VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2019

Michael Jelínek

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

PARAMETRIZACE SÍŤOVÝCH ÚTOKŮ
PARAMETRIZATION OF NETWORK ATTACKS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Michael Jelínek**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Petr Blažek**
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Informační bezpečnost**
Ústav telekomunikací

Student: Michael Jelínek ***ID:*** 195154
Ročník: 3 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Parametrizace síťových útoků

POKYNY PRO VYPRACOVÁNÍ:

Bakalářská práce je zaměřena na definování parametrů identifikujících síťové útoky. Cílem bakalářské práce je definovat alespoň šest parametrů, které budou sloužit k detekci zvolených síťových útoků. Dílčím cílem práce je návrh neuronové sítě (NS), kde vstupem budou definované parametry. V teoretické části práce nastudujte zvolené síťové útoky a definujte metody jejich detekce. V praktické části realizujte NS a implementujte definované parametry. Výstupem bakalářské práce bude systém, který bude schopen na základě apespoň šesti vybraných parametrů detekovat minimálně šest různých anomálií (útoků) v síťové komunikaci.

DOPORUČENÁ LITERATURA:

[1] VONDRÁK, Ivo, 1998. Umělá inteligence a neuronové sítě. Ostrava: Vysoká škola báňská - Technická univerzita Ostrava. ISBN 80-707-8259-5. Dostupné z: http://vondrak.cs.vsb.cz/download/Neuronove_site.pdf

[2] ALI, Siti Hajar Aminah, Seiichi OZAWA, Tao BAN, Junji NAKAZATO a Jumpei SHIMAMURA, 2016. A neural network model for detecting DDoS attacks using darknet traffic features. 2016 International Joint Conference on Neural Networks (IJCNN) [online]. IEEE, 2979-2985 [cit. 2017-09-13]. DOI: 10.1109/IJCNN.2016.7727577.

Termín zadání: 1.2.2019 ***Termín odevzdání:*** 27.5.2019

Vedoucí práce: Ing. Petr Blažek
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Bakalářská práce se věnuje definování vhodných parametrů pro identifikaci síťových útoků za pomoci neuronových sítí. V teoretické části jsou rozebrány metody detekce anomálií v síťové komunikaci, struktura umělé neuronové sítě a DDoS útoky použité pro ověření detekčních schopností. Praktická část se zaměřuje na postup přípravy dat, jejich následnou implementaci do neuronové sítě a shrnutí dosažených výsledků při různě nastavených vlastnostech neuronové sítě.

KLÍČOVÁ SLOVA

DDoS, Anomálie, Útoky, Detekce, Neuronové sítě

ABSTRACT

This bachelor thesis is dedicated to the definition of suitable parameters for network attack identification with the use of neural networks. In the theoretical part of this thesis are methods for anomaly detection in network communication, structure of artificial neural networks and DDoS attacks used for verification of detection capabilities. The practical part of this thesis is focused on the process of preparing data, the subsequent implementation into neural networks and a summary of the results achieved for the different setups of neural networks.

KEYWORDS

DDoS, Anomalies, Attacks, Detection, Neural networks

JELÍNEK, Michael. *Parametrizace síťových útoků*. Brno, 2019, 51 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Petr Blažek,

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Parametrizace síťových útoků“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Petru Blažkovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

podpis autora

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této bakalářské práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno ....

podpis autora

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum a vývoj
pro inovace

Obsah

Seznam obrázků

Seznam tabulek

Úvod

S nárůstem digitalizace a stále masivnějším rozšiřováním informačních technologií do více odvětví obchodu a průmyslu, přibývá hrozeb, které dokáží napáchat velké finanční nebo datové ztráty. Na úkor tomuto faktu musíme být schopni odhalovat a řešit takové hrozby v co nejkratším možném čase.

Ve většině případů je hrozba na síti indikována jako anomálie, která se určitým způsobem vymyká standardním průběhům komunikace. Ve většině případech je anomálie indikátorem útoku, ale ne vždy tomu tak musí být. Může jít např. o důležitou změnu ve vnějším světě, která způsobí zvýšený přístup hostů do sítě. Abychom byli schopni danou hrozbu rozpoznat od legitimního provozu, musíme zvolit vhodné parametry, které nám ji pomohou odhalit. Parametry je nutné volit s ohledem na typ hrozby, před kterou se chceme chránit a také místem nasazení ochrany. Díky velkému množství zařízení připojených do sítě jsou jedním z nejčastějších hrozeb právě útoky typu DDoS, které ve většině případů využívají velkého množství přístupů, aby vyčerpaly kapacitu serveru a všichni ostatní uživatelé museli čekat na uvolnění. Ne však každý útok typu DDoS využívá právě mnoha zařízení k odepření služby. Můžeme se setkat i s útoky *Low and Slow*, které vyčerpávají veškeré kapacity serveru posíláním malých zpráv dostatečně pomalu, aby nedošlo k odpojení od oběti.

Jak je vidět, každý z útoků se chová jinak, a proto je zapotřebí volit vhodné parametry pro jejich odhalení. Právě těmi se zabývá tato bakalářská práce. Vhodně zvolené parametry budou využity k implementaci neuronové sítě, která bude schopna odhalovat nelegitimní anomálie na síťovém provozu. Samotná práce se rozděluje na dvě hlavní části teoretickou a praktickou. Teoretická část se zaměřuje na síťové anomálie, metody jejich detekce, strukturu umělé neuronové sítě a v neposlední řadě na popis útoků, které byly použity pro ověření detekčních schopností výsledné neuronové sítě. Praktická část je zaměřena na přípravu dat, do které patří i volba vhodných parametrů a následné testování neuronové sítě.

1 Anomálie

Nejprve je vhodné říct, co je vlastně myšleno pojmem anomálie. Obecně by se dala popsat jako abnormální, neobvyklé nebo jiné než běžné chování, jak popisuje [1]. Detekce anomálií má široký rozsah využití, ať už se jedná o detekování podvodu s kreditními kartami, detekování narušení síťové bezpečnosti nebo sledování vojenských aktivit nepřítele. Příkladem anomálie narušení bezpečnosti sítě může být neobvyklý datový tok, který může znamenat, že se v síti nachází napadená stanice, jenž odesílá citlivá data neautorizovanému uživateli.

Detekovat takové chování je velice náročné hlavně z důvodu výskytu „falešných poplachů“ způsobených nárazovou změnou proměnných, která však s útokem nemusí mít nic společného. Mezi „falešné poplachy“ mohou patřit změny v chování systému, chyba přístroje nebo i lidský faktor. Většinu anomálií způsobuje právě člověk. Může jít o nechtěnou lidskou chybu, která způsobí neobvyklý výkyv v datech nebo i legitimní chování, kdy nárazově přistoupí na stránku velké množství uživatelů, například kvůli právě probíhající slevě. Taková situace bude z pohledu administrátora vyhodnocena jako anomálie, která může být zaměněna se záplavovými útoky typu DDoS.

Systémy, které se snaží odhalovat anomálie ve formě nelegitimního provozu, musí mít stanoveny určitý model normálního chování. Podle [2] se jedná o formální model, který popisuje vztahy mezi základními proměnnými. Následně vyhodnotí událost jako neobvyklou, pokud její odchylka bude dostatečně velká od stanoveného modelu normálního chování.

Detekce anomálií může být prováděna pomocí různých metod založených na rozdílných principech. Jednotlivé metody budou popsány v následující kapitole. Ne však každá metoda detekce je aplikovatelná na každý problém, proto je dobré kontrolovat kvalitu těchto metod. Nejčastějším způsobem, jak zkontrolovat kvalitu detekční metody, je matice záměn (*confusion matrix*), zobrazená v tab. 1.1, kde odhad i skutečnost, zda se jedná o anomálii, mohou nabývat hodnot *Ano* nebo *Ne*. Z výsledků matice jsou následně vyvozeny další vztahy pro posuzování přesnosti detekce. Nejčastěji se jedná o výpočet přesnosti, který je použit pro demonstraci výsledků této práce.

Tab. 1.1: Matice záměn

		Odhad	
		Ano	**Ne**
Skutečnost	**Ano**	*TP*	*FN*
	Ne	*TN*	*FP*

2 Metody detekce anomálií

Metod, které nám umožňují detekovat anomálie, je celá řada a není výjimkou, když některá metoda využívá více než jeden přístup pro detekci. V následujících kapitolách budou jednotlivé metody blíže popsány.

2.1 Detekce založena na klasifikaci

Klasifikace se využívá k učení modelu z předem označkovaných dat (trénovacích) a následné klasifikace testovacích dat do skupin podle naučeného modelu. Předpokladem pro fungování detekce pomocí klasifikátoru je skutečnost, že klasifikátor je schopen rozlišovat mezi normálními třídami a anomáliemi v daném prostoru popsaném určitými parametry, jak popisují autoři v [3]. Podle dostupných značek v trénovací množině můžeme detekci podle klasifikátoru rozdělit do dvou kategorií.

Klasifikace, která využívá pro normální data více skupin. Proto musí trénovací množina obsahovat více než jen dvě značky (jednu pro označení normálu a druhou pro anomálii). Při takové formě detekce se klasifikátor učí rozhodovat mezi několika skupinami normálu a zbytkem skupin. Když data při testování neodpovídají ani jedné normálové skupině, jsou prohlášena za anomálii.

Klasifikace, která definuje pouze jednu skupinu normálu. Při této technice se klasifikátor učí hranice kolem normálové skupiny a vše, co se nachází za touto hranicí, je prohlášeno za anomálii. K učení se zde využívá *one-class classification algorithmn*.

Neuronové sítě

Neuronové sítě mohou být využity jak s klasifikací více skupin, tak i v klasifikaci pomocí jedné skupiny. Základní metoda detekce pro více skupin se skládá ze dvou základních kroků. Prvním je učení rozpoznávání různých skupin z trénovacích dat a druhým krokem je přivádění testovacích dat na vstup neuronové sítě. Buď je síť schopna zařadit data do některé z naučených skupin nebo je prohlásí za anomálii. Detekci, pomocí neuronových sítí, použili autoři [4] k identifikaci DDoS útoků a dokázali tak, že její využití je vhodnou volbou pro řešení podobných situací. Jelikož tato práce se zabývá detekcí pomocí neuronové sítě, tak podrobnější popis je uveden v následující 3. kapitole.

Bayesovské sítě

Bayesovské sítě jsou používány pro více skupinovou detekci, jak popisují autoři [5]. Tyto sítě jsou acyklické orientované grafy, které zachycují pomocí hran pravděpodobnostní závislosti mezi náhodnými veličinami. Při trénování se síť učí, s jakou

pravděpodobností budou data přiřazeny jaké skupině. Pravděpodobnost je agregována z přidělených parametrů dat. Stejným způsobem se spočítá pravděpodobnost v testovací fázi a porovná se s naučenými daty. Její hlavní výhodou je, že může být použita i v případě nekompletních dat, díky zachytávání pravděpodobnostních vztahů mezi proměnnými.

Metoda podpůrných vektorů

Metoda podpůrných vektorů *(Support Vector Machines - SVM)* využívá techniky učení jedné třídy, kdy se snaží naučit oblast, která obsahuje trénovací data. Cílem je nalézt vhodnou nadrovinu, která rozdělí prostor příznaků. Optimální nadrovina je ta, která má kolem sebe co možná největší pásmo necitlivosti[1]. Pokud je řešený problém lineárně neseparovatelný, lze využít techniku *kernel transformation*, díky které je možné prostor lineárně rozdělit. Při testování se kontroluje, zda data spadají do vymezeného prostoru a budou tak prohlášena za normální, nebo v opačném případě za anomálii. Metodu podpůrných vektorů využili autoři [6] ve své práci k rozpoznávání obličejů, kdy byl vytvořen jeden vektor vstupních hodnot, který reprezentoval stupně šedi na snímku obličeje.

Metoda založena na pravidlech

Detekce založena na pravidlech se učí pravidla zachytávaná z normálního chování systému. Pokud testovaná data nejsou popsána žádným naučeným pravidlem, jsou prohlášena za anomálii. Základní technika založena na pravidlech se skládá ze dvou kroků. Prvním je učení pravidel z trénovacích dat s použitím jednoho z mnoha algoritmů, které slouží k učení pravidel. Příkladem takových algoritmů jsou RIPPER, Rozhodovací stromy a další. Každé pravidlo má přidruženou hodnotu spolehlivosti, která je úměrná poměru mezi počtem správně kvalifikovaných trénovacích dat podle pravidla a celkovému počtu trénovacích dat, na která se pravidlo vztahuje. Druhým krokem je, nalézt nejvhodnější pravidlo pro každou instanci testovacích dat.

Příkladem systému založeným na pravidlech je systém Snort. Jedná se o open-source program, který porovnává každý paket se seznamem pravidel. Pravidla Snortu identifikují útočné pakety na základě IP adres, čísla portu, typů nebo kódů ICMP a informací obsažených v payloadu. Pravidla jsou rozdělena do skupin podle pravděpodobnosti výskytu a s určenou prioritou. Každé pravidlo definované Snortem je popsáno včetně *false positive* a *false negative* výsledků společně s doporučeným postupem opravy, kdyby došlo k narušení. Každý uživatel může přispět novým pravidlem, pokud identifikuje určitý typ škodlivého provozu, a tak rozšiřovat databázi.

[1] Pruh na obou stranách nadroviny, ve kterém se nevyskytují body.

Autoři v [7] prezentují možnost zlepšení úspěšnosti programu Snort při detekci průzkumných útoků a testuje účinnost původní sady pravidel vůči nově definované.

2.2 Detekce technikou nejbližších sousedů

Předpokladem pro techniku nejbližších sousedů je, že instance normálních dat se budou nacházet v těsné blízkosti svých sousedů, zatím co anomálie budou k nejbližším sousedům dosti vzdálené.

Aby bylo možné techniku nejbližších sousedů použít, je zapotřebí definovat vzdálenost nebo podobnosti mezi dvěma instancemi dat. Vzdálenost (nebo podobnosti) mezi dvěma instancemi může být spočtena různými způsoby. Nejpoužívanějším způsobem měření, podle autorů článku [3], je Eukleidovská metrika, která je reprezentována vektorem vzdálenosti.

Detekce anomálií pomocí nejbližších sousedů může být shrnuta do dvou kategorií:

- techniky využívající vzdálenosti instance dat ke k-tému nejbližšímu sousedovi, jako ukazatel anomálie,
- techniky počítající relativní hustotu zastoupení dat, jako ukazatel anomálie.

Metoda K-nejbližších sousedů

Jednoduchá detekce anomálií založena na k-tém nejbližším sousedovi definuje anomálii podle vzdálenosti ke k-tému nejbližšímu sousedovi z poskytnutého datového prostoru. Lze využít i přístupu počítání sumy vzdáleností ke k nejbližším sousedům, jak uvádí [8], kde k může nabývat libovolných přirozených hodnot od 1 po $n - 1$. Pro detekci musí být následně stanovena mez, která bude sloužit k porovnávání vzdáleností. Pokud dojde k překročení této meze, bude instance dat považována za anomálii.

Pro detekci lze využít i jiné přístupy, kdy se nevyužívají jen vzdálenosti k nejbližším sousedům. V [9] autoři využívají hypergrafy pro určení anomálií z množiny dat, které mají různé datové typy nebo jejich výpočet není v běžném pojetí možný, kvůli různým kategoriím dat. Autoři zde modelují hypergrafy z hodnot stejné kategorie a měří vzdálenost mezi instancemi dat analýzou konektivity grafu.

Použití relativní hustoty

Technika detekce anomálií, která spočívá ve výpočtu hustoty objektů v daném okolí. Jak uvádí [10], tak okolí daného objektu lze definovat následovně:

$$HO = \frac{p}{\pi r^2}, \tag{2.1}$$

kde p značí množství objektů v dané oblasti, jinými slovy počet k nejbližších sousedů. Hodnota r značí poloměr kružnice, která definuje oblast, pro kterou je hustota počítána. Existuje několik přístupů k detekci pomocí relativní hustoty, ale základní úvaha je: *Instance dat, která náleží oblasti s velkou hustotou, je považována za normální. Na druhé straně instance, která náleží oblasti s nízkou hustotou je považována za anomálii.*

Jedním z možných přístupů je definice poloměru r ze znalosti datových instancí, kde r je rovno vzdálenosti mezi instancí a k-tým nejbližším sousedem. Zjistit ze vstupních dat obecně k však není jednoduchý úkol, proto se využívají různé odhady jak stanovit poloměr r bez parametru k. Jedním z možných odhadů je výpočet průměrné hodnoty ze všech vzdáleností mezi každou dvojicí vstupních dat. Pro zmíněné případy je však stále nutné znát detekční mez, aby bylo možné rozlišovat anomálie od normálních dat. Obvykle se testovaná data porovnávají s inverzní hodnotou *HO*, aby výpočet odpovídal základní úvaze.

Při jednoduchém přístupu, který je zmíněn výše, může nastat problém, že se anomálie chybně označí jako normální instance, proto byly vyvinuty sofistikovanější přístupy detekce za pomoci relativní hustoty, jako jsou *Local Outlier Factor (LOF), Connectivity-base Outlier Factor (COF)* nebo *Outlier Detection using Indegree Number (ODIN).*

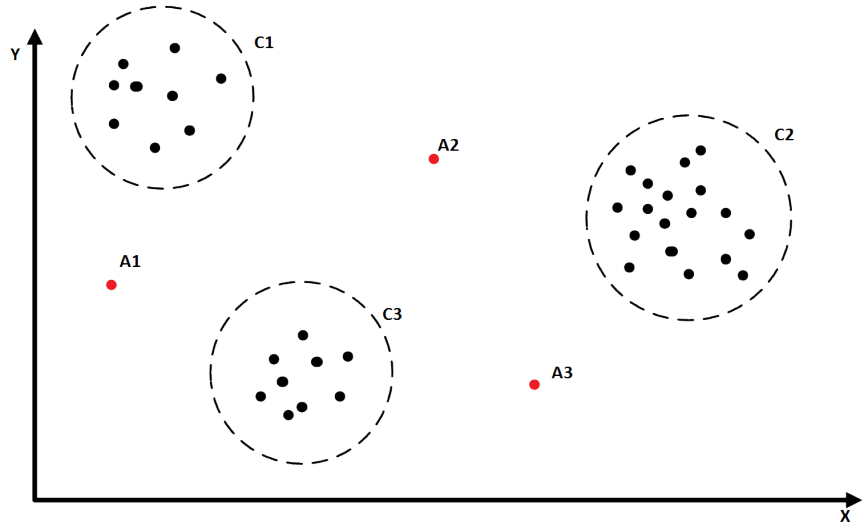
2.3 Detekce založena na klastrovacím modelu

Klastrování, neboli shlukování stejných dat do skupin (klastrů). Primárně se jedná o techniku, která využívá učení bez učitele tzn. sama tvoří referenční model, jak popisuje [11]. Detekce pomocí shlukování se dá rozdělit do tří přístupů.

První přístup operuje s předpokladem, že normální data náleží jednomu ze shluků, zatím co anomálie nebudou patřit ani do jednoho z nich, jak je vidět na Obr.2.1. Nevýhodou prvního přístupu je právě neoptimálnost k nalézání anomálií, protože hlavním cílem algoritmu je nalézt shluky, ne však anomálie.

Druhý přístup uvažuje, že normální data leží blízko středu shluku, zatím co anomálie se budou nacházet daleko od středu shluku. Techniky s tímto přístupem se ve většině případů skládají ze dvou kroků. Prvním krokem je rozřazení dat do shluků pomocí shlukového (klastrovacího) algoritmu. Ve druhém kroku se počítá vzdálenost každé datové instance od středu nejbližšího shluku. Z vypočtené vzdálenosti se zjistí, zda instance dat náleží do shluku, nebo ne. Nejznámějším algoritmem je K-means. Autoři [12] ve své práci tento přístup využívají právě pro detekci v IDS. Další využití nachází v detekci podvodů nebo chyb.

Podle posledního přístupu normální data náleží do velkých a hustých shluků, na druhé straně anomálie spadají do shluků malých a s malou hustotou.



Obr. 2.1: Klastrování bodů ve 2D, kde C značí klastry a A anomálie.

Techniky založené na posledním přístupu tedy stanový instanci za anomálii, pokud její velikost nebo hustota klesne pod určitou mez. Tento přístup je využíván v práci [13] pro detekci anomálií na síťových kartách.

2.4 Detekce založena na statistice

Detekce anomálií pomocí statistiky obvykle vytváří model obecného pravděpodobnostního modelu a následně porovnává, zda testovaná data odpovídají. Podobně jako detekce založena na klasifikaci se skládá ze dvou kroků. Prvním je trénování, kdy je vytvářen statistický model. Druhým krokem je testování, kdy se instance dat porovnává s modelem. Pokud data odpovídají modelu, jsou prohlášena za normální v opačném případě budou prohlášena za anomálii.

Detekce založená na statistice má několik výhod. Jednou z nich je právě schopnost detekce, bez předem definování bezpečnostních pravidel nebo samotných útoků. Dále dokáží poskytovat přesné informace o škodlivých aktivitách za delší časový úsek. Na druhou stranu je velice náročné určit správnou hranici tak, aby pravděpodobnost *false positive* a *false negative* byly v rovnováze. Mimo jiné je zapotřebí použít přesné statistické rozložení, kdy však nastává problém, že nelze určit přesný model pro všechna chování jen za pomoci statistických metod.

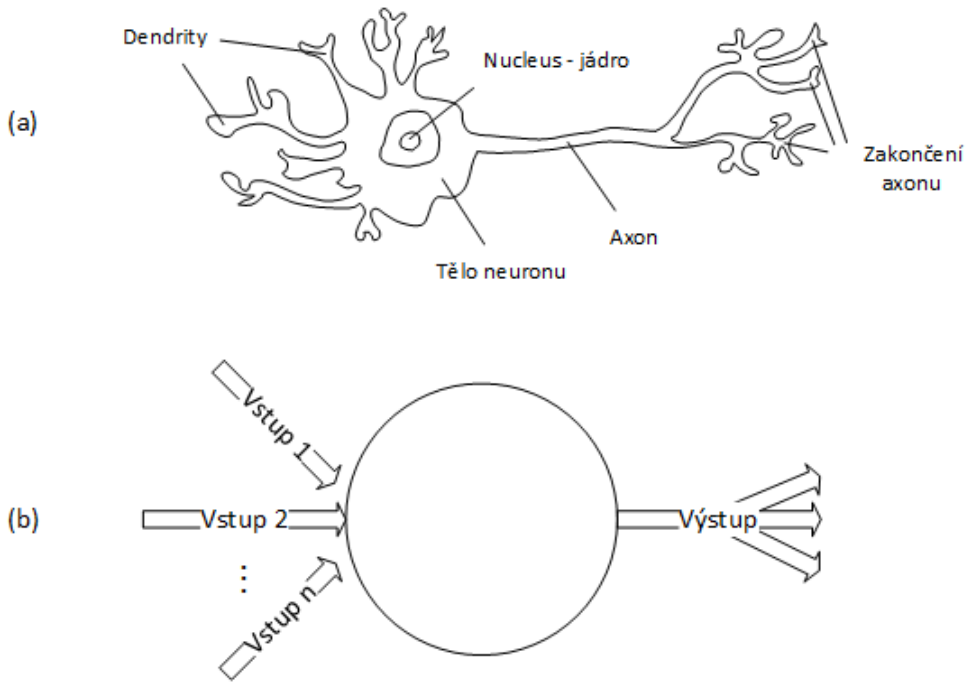
Jedním ze systémů založených na statistických metodách pro detekci paketových anomálií je SPADE [14]. Jako jeden z prvních systémů používal hodnocení odchylek pro detekci skenování portů.

3 Neuronová síť

Jedná se o výpočetní systémy, které se inspirují biologickými neuronovými sítěmi. Snahou je získat správný výstup bez programování specifických pravidel k řešení problému, jak popisuje autor v [15]. Stejně jako biologická neuronová síť, tak i ta umělá je tvořená množstvím neuronů, které jsou propojeny synapsemi, jenž se starají o přenos signálu zpracovaného neuronem. Hlavní vlastností neuronové sítě je schopnost učit se, tzn. schopnost měnit se takovým způsobem, aby bylo dosaženo požadovaných výsledků. Každé spojení mezi jednotlivými neurony je reprezentováno váhou, která se může měnit a tím ovlivňovat výsledky.

3.1 Umělý neuron

Z obr. 3.1 je patrné, že dendrity reprezentují vstupy, nucleus je samotný neuron a zakončení axonu je výstupem (výstupů může být více). Pokud se bavíme o neuronové síti, tak výstupy (axony) neuronu budou napojeny na vstupy (dendrity) následujícího neuronu.

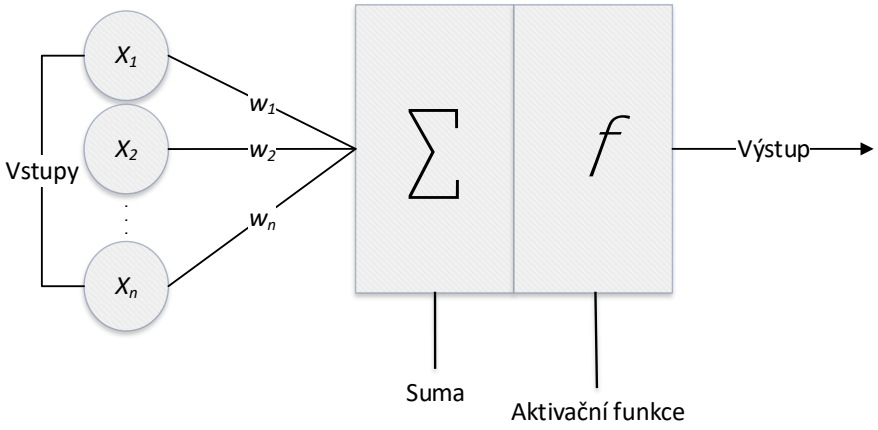


Obr. 3.1: Rozdíl mezi biologickým (a) a umělým modelem neuronu (b).

Každý umělý neuron se skládá ze sumarizační funkce, která sečte vstupy násobené váhami a nakonec přičte b (bias)[1]. Sumarizační funkce má vzorec

$$Y = \sum_{i=1}^{n} (x_i \times w_i) + b, \tag{3.1}$$

kde x značí jednotlivé vstupy a w váhy. Výsledek předá aktivační funkci, jak zobrazuje obr. 3.2, jejímž hlavním úkolem je konverze vstupního signálu na signál výstupní, jak popisuje autor v [15]. Ten je použit jako vstup na další vrstvě.



Obr. 3.2: Vnitřní struktura neuronu.

Aktivačních funkcí existuje několik různých druhů. Mezi nejpoužívanější patří

- Sigmoid – Aktivační funkce má tvar

$$\mathrm{f}(x) = \frac{1}{1 + e^{-x}}. \tag{3.2}$$

  Je velice snadno aplikovatelná, ale mezi její nevýhody patří pomalá konvergence, dále střed funkce není na nule, tudíž postupná aktualizace gradientu způsobuje problematickou optimalizaci.
- Tanh – Funkce hyperbolický tangens má tvar

$$\mathrm{f}(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}. \tag{3.3}$$

  Její rozsah je mezi -1 a 1, proto lze optimalizovat lépe než funkce Sigmoid, ale stejně tak trpí problémem mizejícího gradientu.

[1]Bias – prahová hodnota

- ReLu – Až 6x rychlejší než aktivační funkce Tanh a má tvar

$$R(x) = max(0, x). \tag{3.4}$$

  Předchází problému s mizejícím gradientem, ale mezi její limitace patří zařazení pouze do skryté vrstvy neuronové sítě. Do výstupní vrstvy musí být zařazena funkce Softmax nebo lineární funkce, podle typu testovaných dat. Velkým nedostatkem ReLu je, že vzhledem k použitým trénovacím datům můžou vznikat tzv. mrtvé neurony, které se i po aktualizaci vah nikdy znovu neaktivují.

3.2 Učení neuronové sítě

Učení má za cíl nastavit váhy tak, aby pro určitý vstup dat generovaly správnou odpověď. Po dokončení učení, jehož správnost si můžeme ověřit pomocí statistického měření, by se výsledná neuronová síť mohla nasadit do provozu jako black-box[2]. K učení neuronových sítí se využívá dvou rozdílných přístupů, a to učení s učitelem nebo učení bez učitele, jak popsala autorka v [16].

3.2.1 Učení s učitelem

Na začátku jsou data rozdělena na trénovací, testovací a validační množinu (nejčastěji 50-25-25, nebo 70-15-15, jak popisuje [17]). Trénovací (validační) data je nutné označkovat, aby síť mohla na konci zjistit, s jakou odchylkou se mýlila a pomocí algoritmu Backpropagation zpětně změnit hodnoty vah. Dalším krokem je otestování kvality modelu na testovací množině. Tato práce využívá právě učení s učitelem.

3.2.2 Učení bez učitele

Algoritmus nevyužívá žádné pomoci ve formě předchozího označkování nebo rozdělení do tříd. Sám si je může na základě podobností a rozdílů rozdělovat do skupin, ale ve většině případů je učení bez učitele spojováno s generativními algoritmy. Bližší vysvětlení je nad rámec této práce.

3.2.3 Přeučení sítě

Nastává ve chvíli, kdy se síť naučí příliš dokonale trénovací množinu včetně všech nepodstatných detailů a ztrácí tak schopnost generalizace. To může být způsobeno definicí velkého množství vstupních parametrů na relativně málo pozorováních, ale i implementací většího množství neuronů ve skryté vrstvě, než by bylo potřeba.

[2]Black-box – zařízení nebo jev, u kterého nevíme, jaké procesy probíhají uvnitř

4 Druhy síťových útoků

Síťová komunikace se skládá z dobře známých protokolů, které nám dovolují možností sítí využívat. Použité protokoly jsou standardy a v důsledku toho je jejich implementace a struktura dobře popsána. Síťové útoky využívají mezer, které se v protokolech nachází, aby dokázaly překonat určitou ochranu nebo způsobit škodlivý následek.

Z pohledu administrátora sítě, na kterou se někdo snaží útočit, se může takové chování jevit jako síťové anomálie, o které se musí rozhodnout, zda je nebo není útokem. Sofistikovanější útoky se snaží o splynutí s běžným provozem, aby nebyly tak snadno odhalitelné a tím znesnadňovaly práci tomu, kdo se jim snaží bránit.

Existuje velká spousta druhů útoků, proto se tato práce zaměřuje pouze na některé z nich.

4.1 DDoS útoky

DDoS útoků existuje více druhů a dělí se způsobem jejich provedení. Každý z nich má však stejný cíl, odepřít službu. DDoS útok je veden větším množstvím počítačů, ať už s vědomím majitelů nebo bez něj. Větší útoky bývají prováděny za pomocí botnetu. Jedná se o síť počítačů napadených škodlivým programem (malwarem), který v nich spouští proces čekající na příkaz útočníka. V dnešní době se hojně využívají zařízení z IoT pro jejich slabé zabezpečení. Útok může být i předem plánovaný, a tak na počítačích běží pouze instrukce s časem a pevně danou adresou útoku. Takový způsob je pro útočníka snažší, protože nevyžaduje žádnou další komunikaci.

4.1.1 Low and slow útoky

DDoS útoky se ve většině případů vyznačují svým masivním počtem paketů, kterými zahlcují svojí oběť. Pro dosažení větších objemů dat se rozsáhle využívají botnet sítě, které výsledný objem dat ještě zvětší. Na druhé straně útoky *low and slow*, využívají slabin v protokolech a mohou i při spuštění z jednoho zdroje dosahovat požadovaných výsledků, tudíž odepření služby.

R U Dead Yet? (R.U.D.Y.)

Řadí se do kategorie *low and slow* útoků, protože jeho princip spočívá v donucení serveru, aby čekal na kompletní odpověď útočníka, která přichází po malých částech.

Útočník zašle na server požadavek HTTP metodou POST, který bude v hlavičce obsahovat informaci o velkém množství obsahu. Útočník následně zasílá pakety

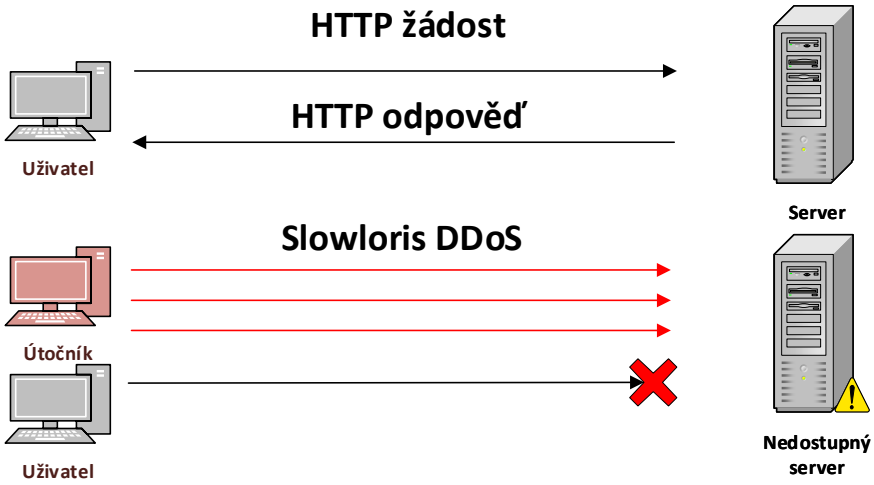
malé velikosti, typicky jeden byte, v intervalu kolem deseti vteřin. Interval musí být vhodně zvolen, protože kdyby byl moc dlouhý, mohlo by dojít ke zrušení čekání na zbytek zprávy. Pokračováním posílání podobných požadavků, dosáhne útočník vyčerpáním kapacit serveru a v důsledku toho zabrání přístupu legitimních uživatelů na server.

V závislosti na průběhu útoku je velice těžké jej odhalit, protože může být snadno zaměnitelný s legitimním provozem. Může nastat situace, že přístup uživatele s pomalým připojením bude vyhodnocen jako nelegitimní a bude mu odepřen přístup.

Slowloris

Slowloris je stejně jako R.U.D.Y řazen do kategorie *low and slow* útoků. Svým principem dostává svému jménu, protože místo běžného *flood*[1] útoku zasílá dotazy HTTP, které donutí server otevřít komunikační kanál a čekat na data od útočníka. Tato data však nikdy nedorazí, proto po uplynutí přednastavené serverové hodnoty timeout se otevřené kanály začnou zavírat. Útočník se snaží udržet kanály otevřené co nejdelší dobu. V případě ukončení komunikace ze strany oběti, posílá další požadavky na otevření nového kanálu.

Každý server má vyhrazenou určitou kapacitu, kterou je schopný poskytovat přistupujícím klientům. Po obsazení všech míst útočníkem nezbývá žádný prostor pro legitimní klienty, kteří se už na server nedostanou. Průběh útoku je zobrazen na obr. 4.1.

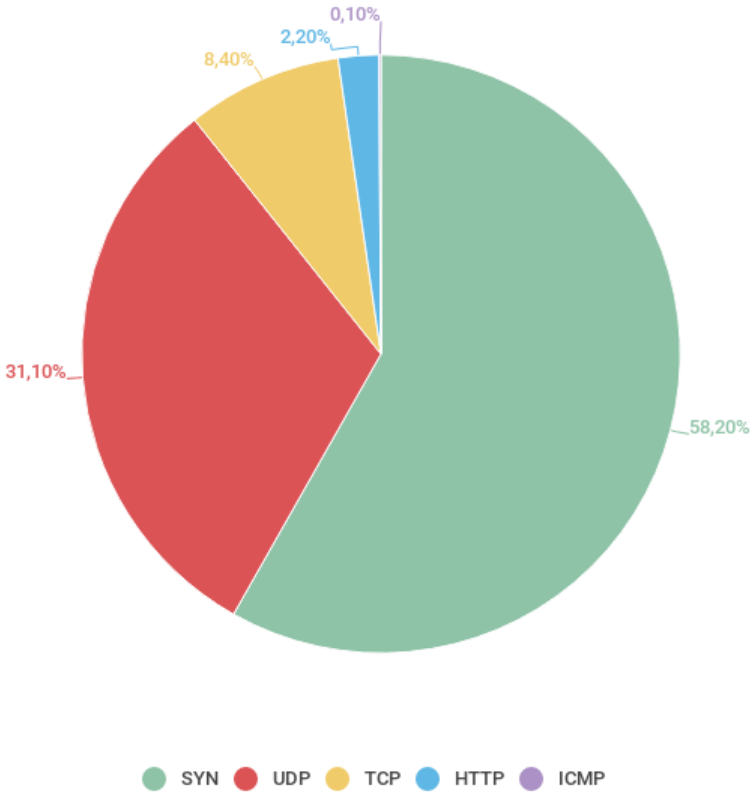


Obr. 4.1: Diagram průběhu útoku Slowloris.

[1] *flood* útoky zahlcují server obrovským množstvím paketů v krátkém čase

4.1.2 Záplavové útoky

Záplavové útoky jsou z kategorie DDoS útoků ty nejvíce využívané, především díky své jednoduchosti. Jejich princip spočívá v zahlcení linky nebo výpočetních prostředků oběti velkým množstvím dat, která způsobí nedostupnost služby. Mezi nejběžnější *flood* útoky ve čtvrtém kvartále roku 2018 patřili SYN a UDP *flood*, které oproti minulým statistikám vzrostly na oblibě. Rozdělení útoků uvádí [18] na obr. 4.2.



Obr. 4.2: Graf rozdělení DDoS útoků v Q4 roku 2018 [18].

SYN flood

Využívá *3-way handshake* v protokolu TCP, který slouží k navázání spojení mezi koncovými stanicemi. Za normálních okolností by klient poslal paket s příznakem SYN na server, který by otevřel spojení a zpátky odeslal paket s příznakem SYN/ACK. Následně by klient odpovídal paketem s příznakem ACK. Útočník při útoku posílá pakety SYN s podvrženou zdrojovou adresou. Server otevře na daném portu spojení a odesílá odpověď SYN/ACK. Následuje čekání na odpověď, která však nikdy

nepřijde, ale server nemůže po určitou dobu spojení zavřít. Při útoku server otevírá množství spojení s útočníkem a vyčerpává tak své kapacity pro legitimní klienty, pro které se stává nedostupným.

UDP flood

Útok pro své provedení využívá transportní protokol UDP, který posílá data bez navázání spojení. Protože není potřeba navazovat spojení a ověřovat komunikující strany, je protokol vhodný např. pro VoIP. Absence navázání spojení však umožňuje podvrhnout zdrojovou adresu a schovat tak identitu útočníka. Samotný útok spočívá v posílání UDP datagramů na náhodné porty oběti. Systém oběti musí nejprve zkontrolovat, zda nějaký program nenaslouchá na daném portu. Pokud ne, tak systém pošle odesílateli odpověď o nedostupnosti.

Operací, které systém oběti musí provést není mnoho, ale díky rychlosti UDP protokolu a velkému množství UDP datagramům dojde k nedostupnosti systému oběti pro legitimní uživatele.

4.1.3 Amplifikační útoky

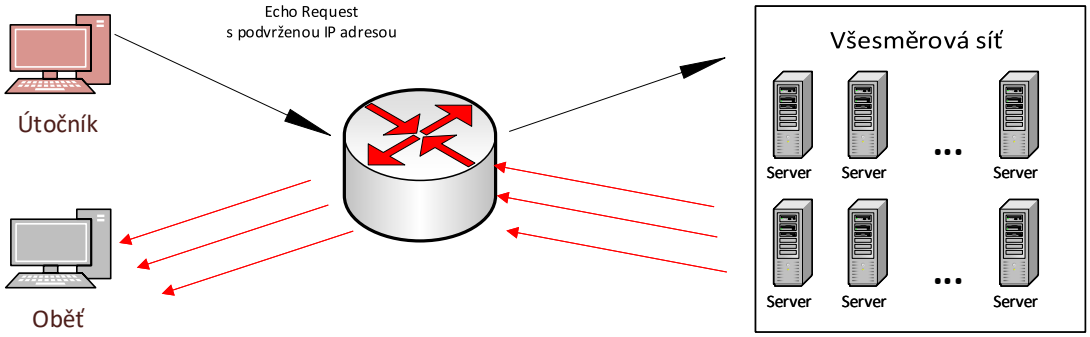
Pro využití amplifikace (zesílení) při síťovém útoku, musí být splněno několik předpokladů. Před samotným odesláním žádosti o poskytnutí dat nesmí proběhnout TCP handshake, který by identifikoval útočníka a znemožnil podvrhnutí zdrojové adresy. Druhým požadavkem je, aby odpověď odeslaná na podvrženou adresu byla násobně větší, než původní dotaz útočníka. Posledním požadavkem pro uskutečnění amplifikovaného útoku je přístup k dostatečnému množství nesprávně nakonfigurovaných systémů. Z toho plyne, že pro komunikaci musí být využitý transportní protokol UDP, který umožňuje změnu zdrojové adresy, a tak data plynoucí z klientské aplikace budou zasílána na cílový server bez dalšího ověřování.

Smurf

DDoS útok, který se snaží o zaplavení oběti ICMP pakety. Útok nejprve rozešle požadavek s podvrženou zdrojovou adresou na směrovač, který má povoleno všesměrové předávání a požadavek rozešle všem zařízením v síti. Každý z dotázaných odpoví na adresu cíle útoku, čímž dojde k mnohonásobnému zesílení záplavy a server se stává nedostupným pro legitimní provoz.

Na serveru je útok dobře rozpoznatelný od běžného provozu, ačkoliv příkaz `PING` se běžně používá pro monitorování dostupnosti zařízení v síti, tak v téhle situaci budou hranice množství příchozích ICMP paketů několikanásobně překročeny a navíc dotazy budou chodit z několika IP adres současně. Může nastat situace, kdy bude

více legitimních klientů posílat ICMP dotazy na server, ale téměř nikdy v takovém množství. Průběh amplifikovaného útoku Smurf je zobrazen na obr. 4.3.



Obr. 4.3: Diagram průběhu útoku Smurf.

NTP amplifikace

NTP amplifikace využívá protokol NTP z rodiny protokolů TCP/IP, který slouží k synchronizaci času mezi počítači a dalšími zařízeními v síti. Jelikož využívá transportní protokol UDP, útočník při zasílání dotazu na server, který bude využit pro amplifikaci, podvrhne zdrojovou adresu za adresu oběti. Server je na útok zranitelný, pokud má povolený příkaz *monlist*, který slouží k monitorování provozu na serveru. Zda je server zranitelný, lze zjisti pomocí příkazu `ntpdc -n -c monlist IP`, kde IP značí adresu serveru. Pokud od serveru dostaneme odpověď, lze daný server využít k NTP amplifikaci. Příkaz *monlist* ve své odpovědi může uvádět až 600 posledních adres, které byly připojeny k serveru.

DNS amplifikace

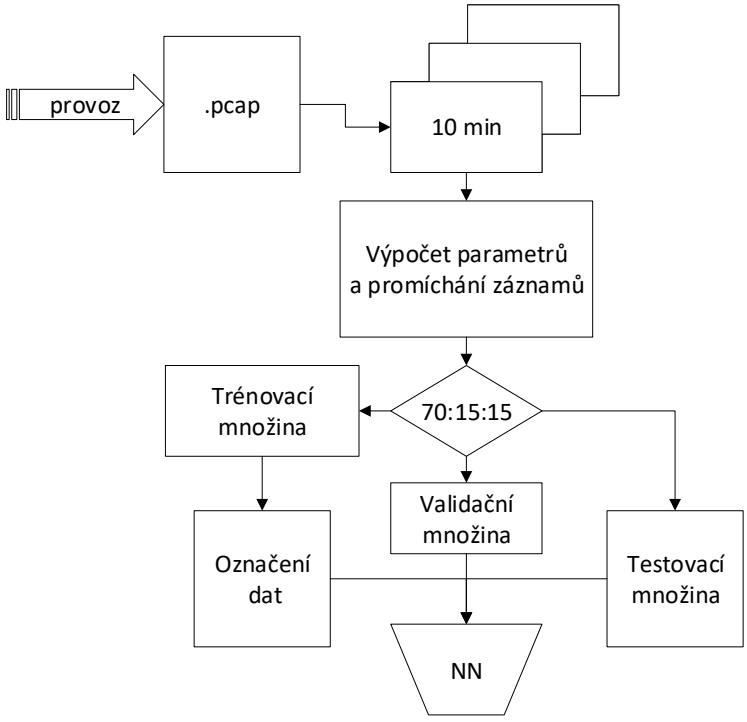
DNS amplifikace zneužívá servery s otevřenými resolvery, na které útočník posílá dotaz typu *ANY* a server mu vrací odpověď, která je několikanásobně větší. Stejně jako NTP amplifikace využívá jako transportní protokol UDP, proto je možné odpovědi na dotaz směrovat přímo na zařízení oběti a vytížit tak její linku, což způsobí nedostupnost pro ostatní uživatele.

5 Příprava dat

V následující kapitole jsou popsány jednotlivé kroky procesu přípravy dat. Aby bylo dosaženo dobrých výsledků, je důležité mít dostatečné množství dat ze síťového provozu, která budou vhodně zpracována pro použití v neuronových sítích. Příprava dat se skládat ze dvou hlavních částí, kterými jsou zpracování dat a výpočty parametrů.

5.1 Zpracování dat

Na obr. 5.1 je vidět, jak probíhá samotná příprava dat před aplikací do neuronové sítě.



Obr. 5.1: Proces přípravy dat.

Postup přípravy je popsán následovně:

- v prvním kroku je důležité zachytávat komunikace, kde bude obsažen jak normální, tak i provoz s útoky. Výsledný soubor je ve formátu `.pcap` a obsahuje všechny pakety, které procházely daným rozhraním v době zachytávání.
- Dalším krokem je rozdělení získaného souboru ve formátu `.pcap` na soubory obsahující komunikaci o délce deseti minut a jejich uložení ve formátu `.csv`

- Následně se vytvoří jeden soubor, který bude obsahovat záznamy s vypočítanými parametry z každého desetiminutového souboru a bude indexován podle zdrojové IP adresy. Počítané parametry jsou popsány v kapitole 5.2. Po dokončení výpočtu všech parametrů se obsah souboru promíchá.
- Promíchaný soubor se rozdělí na trénovací, testovací a validační množinu v poměru 70:15:15.
- Označkování záznamů v trénovací množině podle druhu útoku, který daný záznam reprezentuje.
  0 – legitimní provoz,
  1 – útok slowloris,
  2 – útok Rudy,
  3 – útok *SYN flood*,
  4 – útok *UDP flood*,
  5 – útok NTP amplifikací,
  6 – útok DNS amplifikací,
  7 – útok Smurf.

  Tento krok je důležitý pro učení neuronové sítě, která své výsledky porovnává se značkami u daných záznamů a následně pomocí výpočtu odchylky chyby zpětně upraví váhy mezi jednotlivými neurony.

5.2 Výpočty parametrů

Pro správné rozeznání nelegitimního provozu za pomoci neuronové sítě je dle [19], [20] a [21] klíčová volba vstupních parametrů. Vstupní parametry se volí na základě zvolených anomálií (útoků), specifických vlastností charakterizující tyto útoky v síťové komunikaci. V níže uvedených podkapitolách jsou uvedeny parametry charakterizující definované útoky typu (D)DoS.

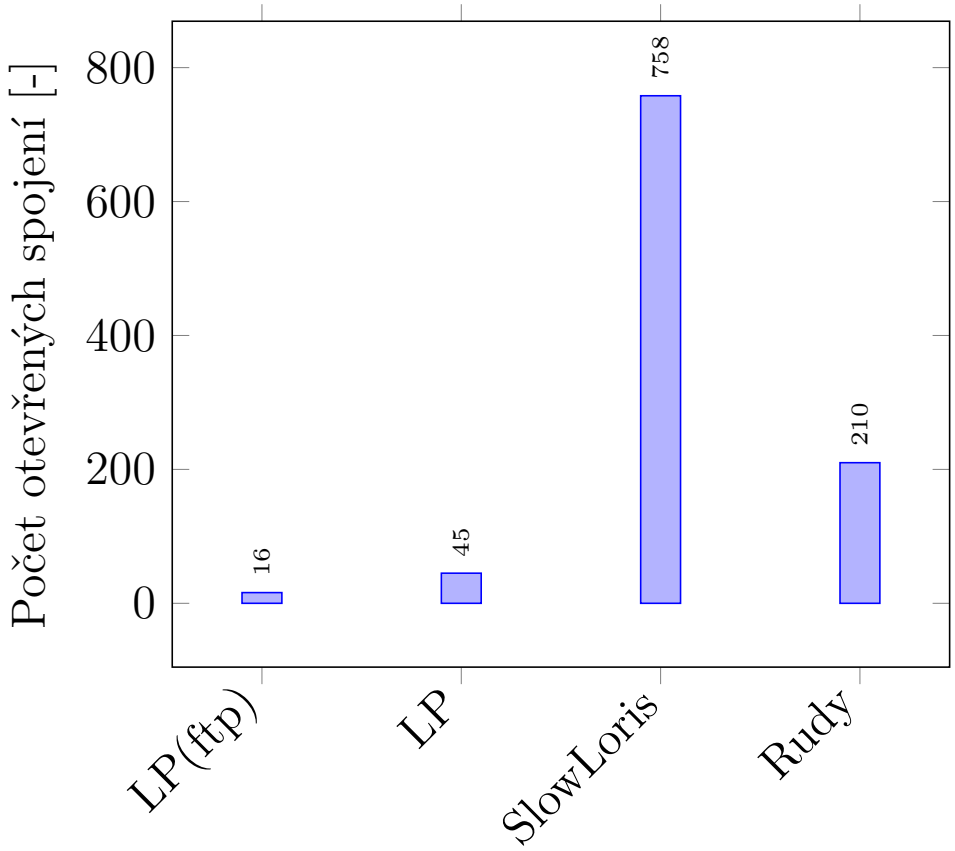
5.2.1 Celkový počet paketů

Prvním zvoleným parametrem je množství všech přenesených paketů za daný časový úsek z jedné zdrojové IP adresy. Typicky DDoS útoky typu *flood* jsou reprezentovány velkým množstvím přenesených paketů za krátký časový úsek. Na druhé straně útoky typu *low and slow* reprezentuje přesně dané množství paketů. I když jsou útoky dobře definovatelné množstvím paketů, může se stát, že i během legitimního provozu, který není pod útokem zvenčí, bude množství paketů výrazně kolísat. Funkce, která v kódu zpracovává tento parametr, dostane na vstup všechny pakety z jedné IP adresy a vrací jejich počet.

5.2.2 Počet otevřených TCP spojení

V téměř každém síťovém provozu se navazuje množství TCP spojení. Není však úplně běžné otevřít z jedné zdrojové adresy velké množství spojení během krátkého časového úseku. Útoky popsány v kapitole 4.1.1 jsou ukázkovým příkladem. Principem jejich průběhu je otevřít značné množství spojení a více, či méně je udržovat otevřené po určitou dobu. Protože cílem téhle práce je detekovat právě zmíněné útoky, tak tento parametr hraje velkou roli při jejich odhalování.

TCP protokol používá *3-way handshake*[1] k navázání spojení. Počet navázaných spojení z jedné adresy bude detekováno počítáním navázaných spojení tzn. spojení je navázáno pokud na TCP paket *SYN* dostaneme odpověď *ACK*. Na obr. 5.2 lze vidět patrný rozdíl mezi útoky a legitimním provozem (LP), kdy útoky navazují spojení v řádu stovek za minutu a legitimní provoz pouze v řádu desítek. Přičemž první sloupec značí komunikaci z FTP serveru a druhý sloupec komunikaci zachytávanou na uživatelské stanici při běžné práci.



Obr. 5.2: Počet otevřených spojení za 60s.

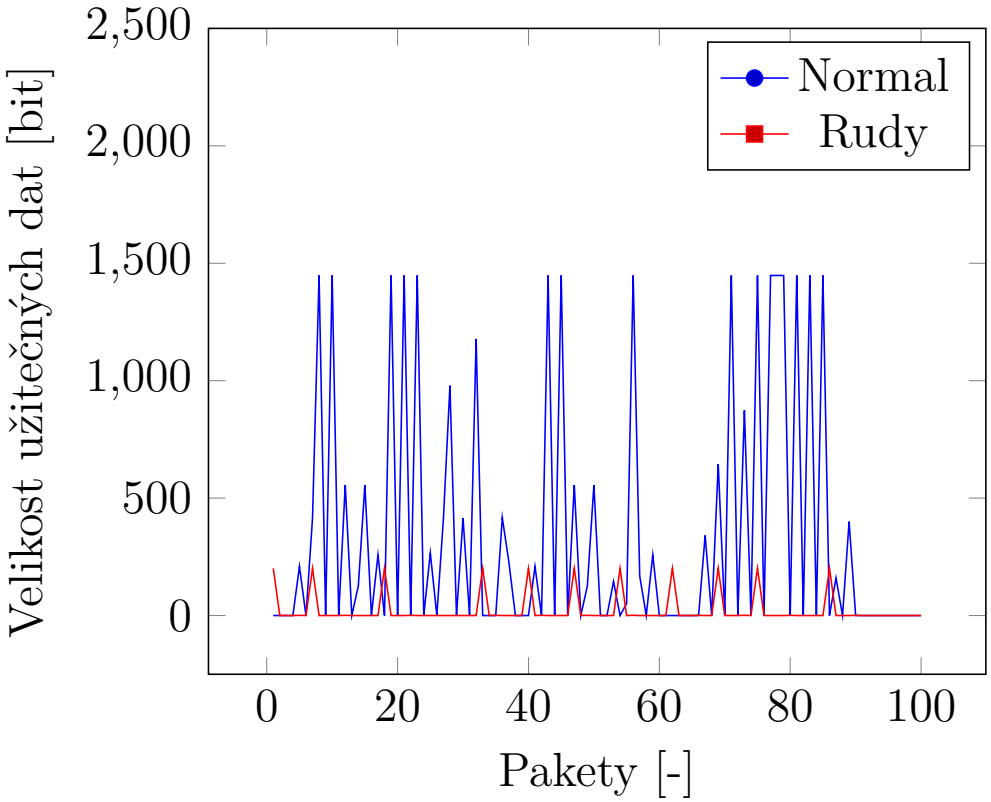
[1] Trojice zpráv SYN, SYN/ACK, ACK sloužící k navázání spojení.

5.2.3 Průměrná velikost užitečných dat

Útoky, které posílají velké množství dat po síti, bývají snadno odhalitelné, ale na druhé straně útoky, které jsou charakterizovány minimální velikostí užitečných dat, bývají snadno zaměněny s legitimním provozem. Mnohdy se stává, že velikost užitečných dat může být nulová. Avšak i takový případ může identifikovat určitý druh anomálie a tím přispět k odhalení útoku. Na obr. 5.3 je vidět patrný rozdíl ve velikosti užitečných dat při běžné komunikaci a útoku. Legitimní komunikace se liší především ve střídání různých velikostí užitečných dat na rozdíl od útoku, který je reprezentován průběhem se stále se opakujícími hodnotami. Vzorec pro výpočet je popsán následovně

$$\mathrm{PVD} = \frac{\sum_{n=1}^{i} U_i}{P}, \tag{5.1}$$

kde U značí užitečná data a P počet paketů s užitečnými daty z jedné IP adresy.



Obr. 5.3: Porovnání velikosti užitečných dat při útoku a běžném provozu.

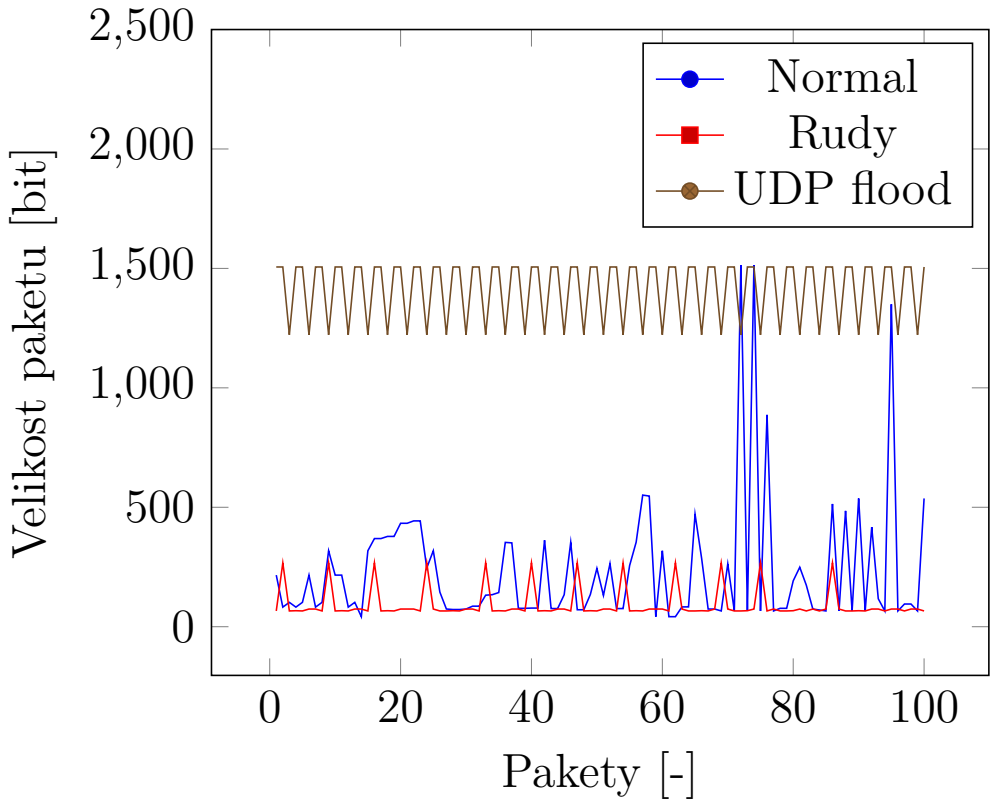
5.2.4 Průměrná velikost paketu

Velikost paketu se od 5.2.3 značně liší. Hlavním rozdílem je, že velikost paketu nebude nikdy nulová. Na rozdíl od 5.2.3, kdy velikost nemusí být udávána vůbec.

Vzorec pro výpočet je popsán následovně

$$\text{PVP} = \frac{\sum_{n=1}^{i} V_i}{P}, \tag{5.2}$$

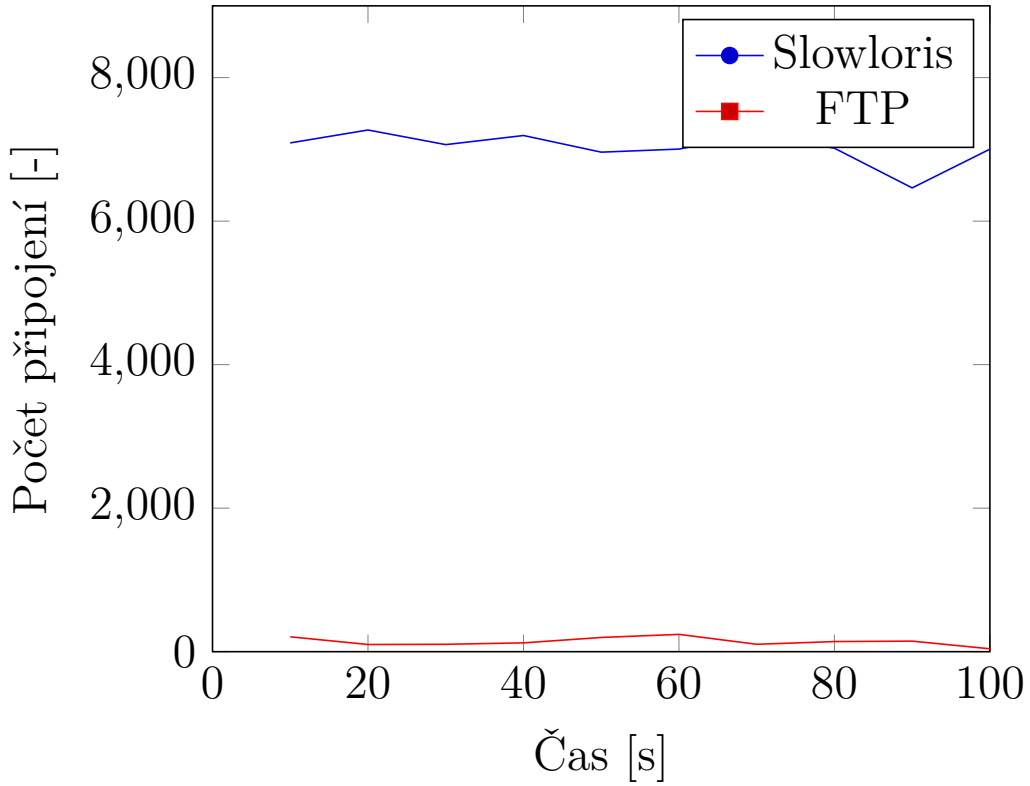
kde V značí velikost paketu a P udává počet všech paketů z jedné IP adresy. Při DDoS útocích typu *flood* se útočníci snaží o zahlcení oběti velkým množstvím paketů v co možná nejkratším čase. Z důvodu urychlení zasílání se volí malá velikost paketů, proto i průměrná velikost paketu bude oproti normálu značně menší. Z obr. 5.4 je zřejmé, že průměrná velikost paketů za daný čas pro útok *UDP flood* bude konstantní a mnohem větší, než při útoku Rudy nebo běžném provozu.



Obr. 5.4: Porovnání velikosti paketu při útoku a běžném provozu.

5.2.5 Počet připojení hosta z různého portu

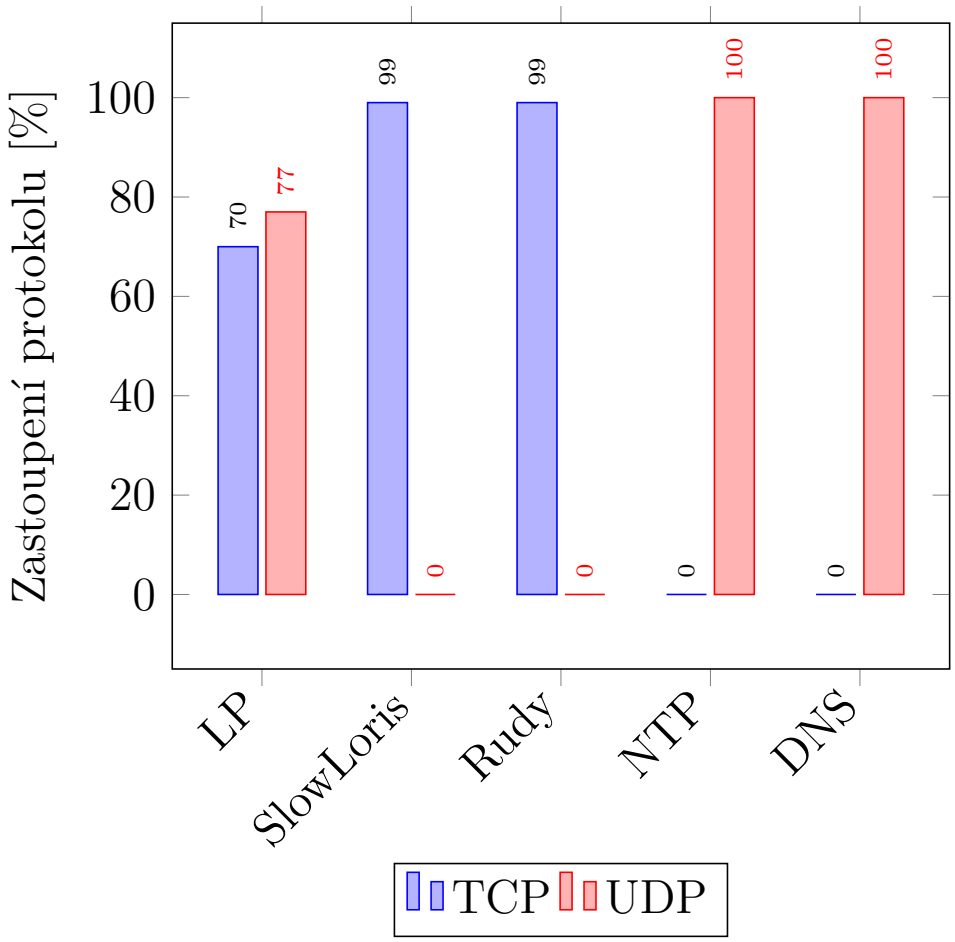
Parametr, který značí počet připojení hosta (IP) z různých portů v časovém úseku desíti minut napomáhá v bližším určování útoků, které jsou méně sofistikované a útočník tak veškerou škodlivou činnost provádí z jedné IP adresy. Pří běžné komunikaci lze takové chování snadno rozeznat, protože u určitých druhů útoků dochází k enormnímu nárůstu opakované snahy o připojení z různého zdrojového portu. Na obr. 5.5 je vidět porovnání počtu připojení z různého portu při útoku SlowLoris a komunikace protokolem FTP, kde se pro legitimní provoz hodnoty pohybují v desítkách připojení, zatímco při útoku jsou v řádech tisíců.



Obr. 5.5: Porovnání počtu připojení z různého portu při útoku a běžném provozu.

5.2.6 Nejvíce zastoupený protokol hostem v procentech

Při běžné komunikaci se z jedné adresy obvykle používá více protokolů, zatím co při určitých typech útoků, jako je například *syn flood*, je využíván pouze jeden. Tudíž jeho zastoupení bude v takovém případě 100%. Může se stát, že i při útoku bude využívat jiný protokol, ale bude to v mnohonásobně menším množství, než při běžné komunikaci. Navíc každý typ útoku využívá specifický typ protokolu, který jej do jisté míry charakterizuje, proto je procentuální zastoupení společně s protokolem užitečný parametr při správné detekci síťových útoků. Na obr. 5.6 lze vidět, že průměrné zastoupení protokolu TCP a UDP se pohybuje v rozmezí od 70% po 80% pro legitimní provoz. Na druhé straně útoky SlowLoris a Rudy dosahují téměř 100% zastoupení protokolu TCP, ale protokol UDP není zastoupen vůbec. Pro útoky DNS a NTP amplifikace je zastoupení opačné.



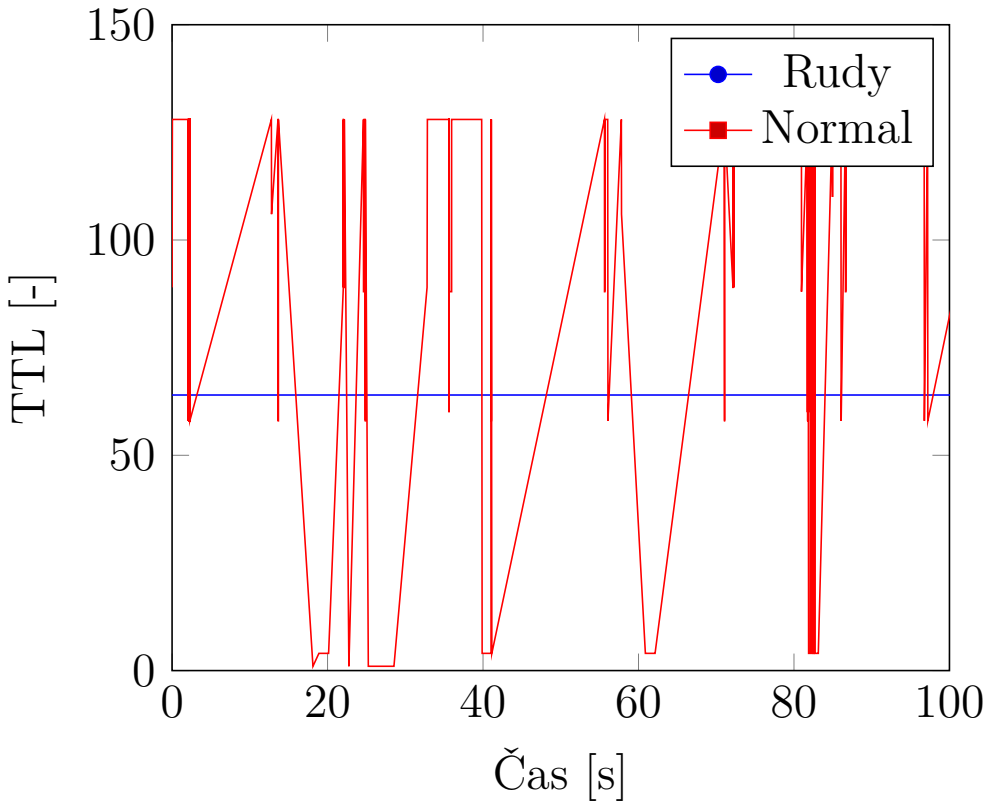
Obr. 5.6: Procentuální zastoupení protokolů při různém provozu.

5.2.7 Průměrná hodnota TTL po dobu 10 min

TTL, neboli *Time to live*, je hodnota obsažena v IP paketech a značí, kolik skoků přes směrovače ještě může daný paket udělat, než dojde k jeho zahození. Velikost *TTL* je volena odesílatelem paketu a v běžné komunikaci je různorodost této hodnoty mnohem větší, než v případě útoků. Při útoku se obvykle jedná o předem definovaný postup příkazů nebo skript a hodnota *TTL* je tak nastavována všude stejně. Porovnání, která zachycuje průběh útoku *Rudy* a legitimní komunikace z pohledu velikosti *TTL*, je zobrazeno na obr. 5.7. Vzorec pro výpočet je popsán následovně

$$\mathrm{PT} = \frac{\sum_{n=1}^{i} T_i}{P}, \tag{5.3}$$

kde T značí hodnotu *TTL* a P značí počet paketů, které obsahují hodnotu *TTL*.



Obr. 5.7: Porovnání průměrné hodnoty TTL při útoku RUDY a běžné komunikaci.

6 Praktická realizace

Cílem této práce je vybrání vhodných síťových útoků, výběr charakterizujících parametrů a následná realizace neuronové sítě, která bude schopna na základně vstupních parametrů rozpoznávat běžný provoz od útoků a ty správně klasifikovat. Pro dosažení, co možná nejlepších výsledků je nutné, kromě volby vhodných parametrů, otestovat neuronovou síť s různým množstvím neuronů a vybrat nejvhodnější aktivační funkci, abychom našli nastavení, se kterým dosahuje nejlepších výsledků při detekci anomálií v síťové komunikaci.

6.1 Příprava a zpracování dat

Dobrý výsledek detekce útoků s využitím neuronových sítí může být dosažen pouze tehdy, pokud bude mít neuronová síť možnost natrénovat dostatečně univerzální model, který bude schopen rozeznat původ dat s vysokou přesností. Proto je nutné ještě před samotným trénováním, získat dostatečné množství dat, která budou reprezentovat všechny skupiny, jenž budeme chtít detekovat. Čím větší množství dat, tím lepší model pro detekci je neuronová síť schopna vytvořit.

Data, která reprezentují normální provoz jsou tvořena ze tří rozdílných množin. V prvním případě se jedná o komunikace uživatelů s FTP serverem, ze kterého byly stahována určitá data v časovém okně 24 hodin. Data byla odchytávána na rozhraní FTP serveru a hranicí vnitřní sítě. Vytížení daného serveru se během dne značně měnilo, což poskytuje neuronové síti lepší podmínky k učení, díky rozdílným výsledkům parametrů za daný čas. Druhá množina obsahuje data přibližně 33 hodin a reprezentuje provoz uživatelské stanice při běžné práci na PC. Obsahem dat je procházení webových stránek, komunikace přes VoIP a streamování muziky. Poslední množina obsahuje pouze komunikaci protokolem DNS a je dostupná z [22]. Protože cílem této práce nebylo nasazení detekce do reálného provozu s určitým typem síťových služeb, tak různorodost legitimních dat povede při trénování sítě k obecnějšímu modelu pro detekci legitimního provozu.

Záznam nelegitimní komunikace byl zaznamenáván ve virtuálním prostředí programu Oracle VM Virtual Box mezi stanicí s operačním systémem Kali-Linux x64 a stanicí se systémem Linux Mint x64, na kterém software Metasploitable x64 zajišťoval vytvoření stránky, která byla následně využita jako cíl útoku Slowloris a Rudy. Samotné útoky Rudy a Slowloris byly generovány aplikacemi Pyslowloris a R-U-Dead-Yet (verze 2.2) dostupnými z [23], [24]. Útoky typu *flood* a amplifikační útoky byly vygenerovány v laboratorním prostředí za pomoci simulátoru Spirent Avalanche 3100B.

Generované data byla ukládána ve formátu *.pcap* a pro další zpracování realizovaného detekčního systému byla převedena do formátu *.csv*. Pro převod formátů byl využit volně dostupný nástroj Wireshark (verze 2.6.4), protože převod pomocí knihovny v použitém programovacím jazyce byl časově mnohem náročnější (s narůstajícím objemem dat exponenciálně narůstala doba zpracování). Další krok přípravy dat je zaměřen na dolováním znalostí, v případě této práce parametrů, ze získaných souborů. Podrobný popis parametrů a schéma postupu zpracování dat se nachází v kapitole 5.

6.2 Neuronová síť

Realizovaná nerunová síť se v implementaci skládá ze tří vrstev. První vrstvou je vrstva vstupní, která obsahuje 8 neuronů. Její jedinou funkcí je vyslat z každého neuronu data do všech neuronů skryté vrstvy, které získaná data zpracují a vyšlou do vrstvy výstupní výsledky. Každé spojení mezi neurony je ohodnoceno vahou, která se během trénování modelu upravuje a dochází tak k tvorbě výsledného klasifikačního modelu. Neuronová síť byla implementována v jazyce `Python 3.7` s využitím knihovny Scikit-learn[1], které poskytují širokou škálu funkcí využitelných při strojovém učení. Neuronová síť vytvořená pomocí této knihovny poskytuje velké množství možností, kterými může být ovlivněna kvalita výsledného měření. Testování vhodného nastavení je popsáno v následující části.

Druhá vrstva se skládá z množství neuronů. Jejich počet můžeme měnit podle velikosti vstupního vektoru a tím ovlivňovat kvalitu výpočtů neuronové sítě. Pro dosažení nejlepších výsledků byla provedena měření přesnosti detekce neuronové sítě s různým množstvím neuronů ve skryté vrstvě a zároveň i vliv aktivační funkce na přesnost a rychlost dosažení výsledku.

6.2.1 Testování počtu neuronů

Počet neuronů ve skryté vrstvě přímo ovlivňuje výsledky dosažené neuronovou sítí, a proto je nezbytné docílit takového množství, při kterém budou výsledky nejlepší. Podle autorů [25] lze množství neuronů pro jednu skrytou vrstvu určit vzorci 6.1 a 6.2. Autor [26] udává vzorec 6.3, jako hraniční množství, při kterém by ještě nemělo docházet k přeučení sítě.

$$N_h = N + 1, \tag{6.1}$$

$$N_h = 2N + 1, \tag{6.2}$$

$$N_h = \frac{N_p}{\alpha(N + N_v)}, \tag{6.3}$$

[1]Dostupná z https://scikit-learn.org/stable/index.html.

kde N_h reprezentuje počet neuronů ve skryté vrstvě. N značí počet neuronů ve vstupní vrstvě, N_v počet neuronů výstupní vrstvy a N_p počet prvků v trénovací množině. Symbol α je označení škálovatelnosti v obvyklém rozsahu 2-5.

Nicméně volba množství neuronů ve skryté vrstvě se odvíjí hlavně od problematiky, kterou má daná síť řešit. Proto je lepší otestovat, při jakém množství neuronů bude použitá neuronová síť dávat nejlepší výsledky a pak nastavení s nejlepšími výsledky použít k dalšímu testování.

Pro testování nejvhodnějšího počtu neuronů ve skryté vrstvě, byla použita všechna dostupná data, tak jak jsou popsána v kapitole 6.1. Pro zjištění, jaké množství neuronů je nejlepší, byly použity následující ukazatele úspěšnosti:

- přesnost – vzorec pro výpočet vychází z matice záměn a výsledná hodnota se vypočítá jako

$$přesnost = \frac{TP + TN}{TP + TN + FP + FN}, \tag{6.4}$$

  kde čitatel označuje počet všech správně označených záznamů a jmenovatel počet všech záznamů (proměnné TP,TN, FP a FN vycházejí z matice záměn 1.1),
- senzitivita (recall) – neboli citlivost, kdy hodnota reprezentuje poměr mezi počtem správně klasifikovaných záznamů a všech pozitivně klasifikovaných. Vzorec je popsán následovně

$$R = \frac{TP}{TP + FN}, \tag{6.5}$$

  kde proměnné TP a FN vycházejí z matice záměn 1.1.
- F-score – bere v úvahu jak přesnost, tak senzitivitu a výslednou hodnotou je jejich harmonický průměr. Základní vzorec se dá použít pouze pokud jsou výsledky binární tzn. klasifikátor určuje buď 0 nebo 1, proto v této práci byla použita hodnota *F1_macro*, která vypočítává hodnoty pro každou třídu klasifikátoru a výsledkem je jejich průměr.

Výsledky všech zmíněných ukazatelů jsou v intervalu od 0 po 1, kde hodnota 1 určuje nejlepší možný výsledek a spodní hranice interval představuje minimální hodnotu pro ukazatele úspěšnosti. V tabulce 6.1 jsou uvedeny výsledné hodnoty pro testování různého počtu neuronů, při použití aktivační funkce *ReLu*.

Z tabulky 6.1 lze vyčíst, že nejlepších výsledků neuronová síť dosahuje s použitím 15 neuronů ve skryté vrstvě neuronové sítě, která má pouze jednu skrytou vrstvu. Proto pro další fáze testování bylo použito právě 15 neuronů.

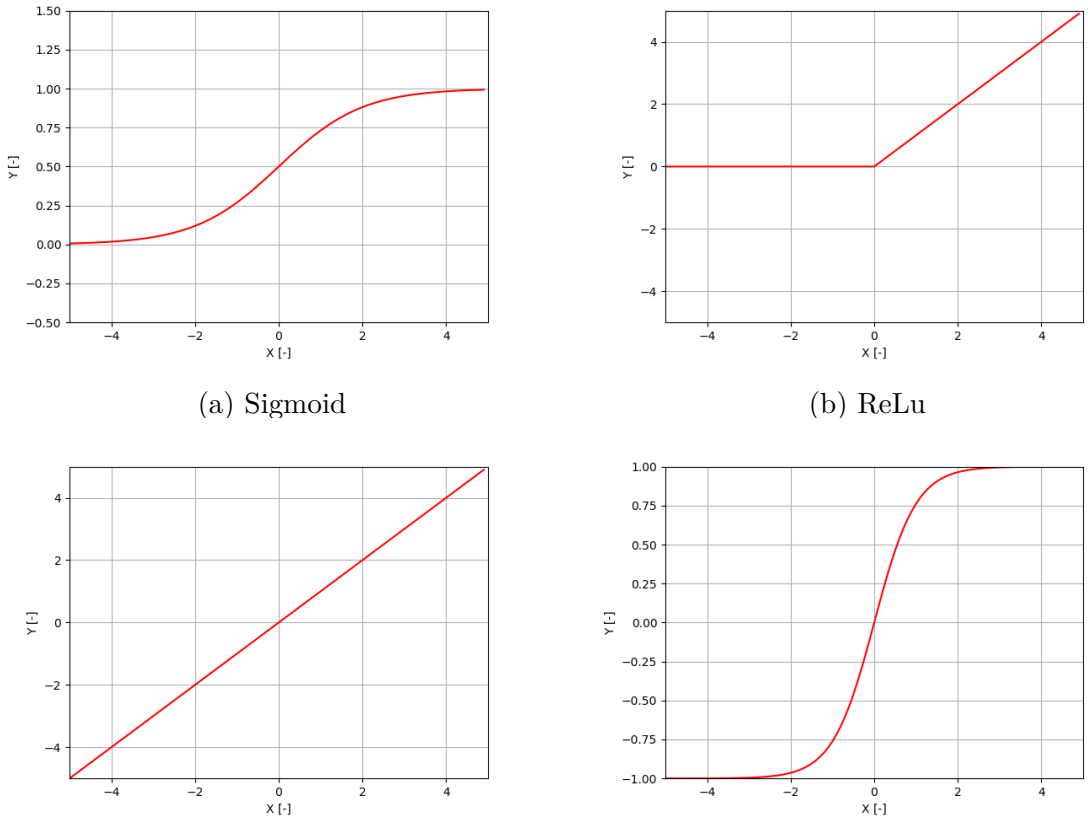
Tab. 6.1: Výsledky testování různého počtu neuronů ve skryté vrstvě.

Počet neuronů [-]	**10**	**15**	**20**	**25**	**30**	**35**	**40**	**45**
Přesnost [-]	0,972	0,978	0,972	0,969	0,972	0,970	0,966	0,970
Senzitivita [-]	0,998	0,999	0,998	0,998	0,998	0,999	0,998	0,998
F1_macro [-]	0,977	0,984	0,979	0,977	0,978	0,979	0,973	0,978

Počet neuronů [-]	**50**	**55**	**60**	**65**	**70**	**80**	**90**	**100**
Přesnost [-]	0,965	0,977	0,970	0,972	0,974	0,975	0,970	0,970
Senzitivita [-]	0,998	0,998	0,998	0,998	0,998	0,999	0,998	0,998
F1_macro [-]	0,973	0,979	0,979	0,982	0,975	0,978	0,980	0,980

6.2.2 Testování aktivační funkce

Volba aktivační funkce se nejčastěji odvíjí od zkušeností z řešení již obdobných úloh a podle povahy vstupních dat. Pro výběr nejlepší aktivační funkce, bylo provedeno měření přesnosti neuronové sítě s využitím čtyř rozdílných aktivačních funkcí. Průběh jednotlivých funkcí je zobrazen na obr. 6.1.



(a) Sigmoid

(b) ReLu

(c) Identita

(d) Tanh

Obr. 6.1: Aktivační funkce

Měření bylo provedeno na dvou vstupních testovacích množinách, kde první reprezentovala malý výběr dat se zastoupením všech testovaných tříd v rozsahu do 1000 záznamů. Druhé měření bylo provedeno na všech datech, jak jsou popsány v kapitole 6.1. V rámci testování aktivační funkce byly testovány i optimalizační algoritmy, které dokáži urychlit nalezení globálního minima odchylky chyby pomocí vhodné úpravy vah spojení mezi jednotlivými neurony. Testovanými optimalizačními algoritmy jsou:

- LBFGS – optimalizační algoritmus z metod quasi-Newton jenž aproximuje BFGS algoritmus, který využívá omezené množství výpočetních prostředků a dokáže rychle dosáhnout požadovaného minima,
- SGD – (Stochastic Gradient Descent) široce používaný optimalizační algoritmus, který upravuje váhy podle průměru všech gradientů. Jelikož průměrováním gradientů dochází k vytváření šumu, může nastat, že nikdy nebude konvergovat k nule, jak popisuje autor [27],
- Adam – na rozdíl od ostatních zmíněných optimalizačních algoritmů, které používají pevnou rychlost učení, tak Adam dokáže rychlost učení měnit v závislosti na gradientu a tím rychleji dosahovat požadovaných výsledků.

Tab. 6.2: Testování funkce Identita.

Ukazatel úspěšnosti	**Velká množina**			**Malá množina**		
	LBFGS	**SGD**	**Adam**	**LBFGS**	**SGD**	**Adam**
Přesnost [-]	0,824	0,716	0,819	0,965	0,8171	0,959
Senzitivita [-]	0,996	0,992	0,996	0,998	0,997	0,994
F1_macro [-]	0,845	0,722	0,838	0,992	0,835	0,985
Čas výpočtu [s]	19	36	41	1	2	2

Tab. 6.3: Testování funkce Sigmoid.

Ukazatel úspěšnosti	**Velká množina**			**Malá množina**		
	LBFGS	**SGD**	**Adam**	**LBFGS**	**SGD**	**Adam**
Přesnost [-]	0,962	0,375	0,947	0,991	0,286	0,588
Senzitivita [-]	0,999	0,971	0,998	0,998	0,839	0,967
F1_macro [-]	0,974	0,370	0,961	0,986	0,270	0,689
Čas výpočtu [s]	10	41	39	1	2	3

Tab. 6.4: Testování funkce Tanh.

Ukazatel úspěšnosti	**Velká množina**			**Malá množina**		
	LBFGS	**SGD**	**Adam**	**LBFGS**	**SGD**	**Adam**
Přesnost [-]	0,979	0,716	0,954	0,991	0,817	0,960
Senzitivita [-]	0,998	0,993	0,994	0,998	0,978	0,996
F1_macro [-]	0,968	0,724	0,974	0,990	0,845	0,989
Čas výpočtu [s]	22	44	38	1	3	3

Tab. 6.5: Testování funkce ReLu.

Ukazatel úspěšnosti	**Velká množina**			**Malá množina**		
	LBFGS	**SGD**	**Adam**	**LBFGS**	**SGD**	**Adam**
Přesnost [-]	0,978	0,716	0,957	0,991	0,816	0,960
Senzitivita [-]	0,999	0,994	0,998	0,998	0,978	0,998
F1_macro [-]	0,982	0,728	0,978	0,985	0,845	0,988
Čas výpočtu [s]	21	41	33	1	2	3

Výsledky zobrazené v tabulkách 6.2, 6.3, 6.4 a 6.5 zobrazují výsledné hodnoty pro jednotlivé aktivační funkce a aplikované optimalizační funkce. Z výsledků vyplývá pro všechny aktivační funkce jako nejlepší volba optimalizační algoritmus *LBFGS*, který prokazuje nejlepší výsledky při klasifikaci i při časovém porovnání s ostatními optimalizačními algoritmy.

Podle dosažených výsledků je pro detekci útoků v síťovém provozu nejlepší volbou aktivační funkce *ReLu*. Všechny tři ukazatele úspěšnosti klasifikace měla nejvyšší ze všech, avšak časově nejvýhodnější se stala aktivační funkce *Sigmoid*. V rámci testování byla dále používána aktivační funkce *ReLu*. Pro nasazení neuronové sítě do provozu by však stálo za zvážení použití aktivační funkce *Sigmoid*, kvůli úspoře času a k přispění rychlejší detekce útoků.

6.3 Detekce anomálií

Detekce anomálií probíhala ve dvou fázích. V první fázi se natrénoval klasifikační model neuronové sítě za pomocí množiny dat, která je popsána v kapitole 6.1. Ve druhé fázi byl zbytek dat použit k získání výsledků pro každou skupinu útoků zvlášť i pro útoky jako celek. Výsledky z testování byly porovnány společně s výsledky dosaženými s použitím cross-validace.

Trénování modelu

Trénovací množina obsahovala 70% všech dostupných dat podrobně popsaných v kapitole 6.1. V této množině bylo zastoupeno 7 druhů útoků vůči sobě v různém množství a legitimní provoz, který reprezentoval komunikaci na FTP serveru a data zachytávaná na běžné uživatelské stanici. V kapitole 6.2.1 a 6.2.2 bylo testováno nejlepší nastavení neuronové sítě pro detekci anomálií v síťovém provozu, ze kterého jako efektivní počet neuronů ve skryté vrstvě vyšel 15. Nejefektivnější aktivační funkcí se ukázala funkce *ReLu* s optimalizačním algoritmem *LBFGS*. Maximální počet cyklů trénování dat (interval, ve kterém dojde k předložení všech záznamů z trénovací množiny alespoň jednou) byl nastaven na 200, ale mohla nastat situace, kdy optimalizační funkce konvergovala k výsledku dříve a učení bylo ukončeno ještě před dosažením maximálního limitu cyklů.

Testování a validace modelu

Testování a validace modelu probíhala s pomocí dat, která nebyla použita pro trénování modelu neuronové sítě. Všechna data byla rozdělena v poměru 70:15:15, kde testovací i validační množina mají zastoupení 15% všech dostupných dat. Celková velikost testovací množiny byla 3303 kombinovaných záznamů legitimního provozu a útoků. Validační množina obsahovala 3274 záznamů. Testování proběhlo ve dvou fázích. První fáze testovala kvalitu sítě pouze na rozpoznávání legitimního provozu a útoků. Ve druhé fázi už proběhlo testování rozpoznávání legitimního provozu a všech použitých typů útoků. Podrobné výsledky včetně porovnání jsou uvedeny v kapitole 6.4.

Validace

Validační množina dat je v procesu učení umístěna mezi trénováním a testováním. Jejím použitím můžeme předcházet přeučení sítě, což by mohlo způsobit vytrácení schopnosti generalizace (zobecnění) modelu, jak je popsáno v kapitole 3.2.3.

Neuronová síť s přeučeným modelem už by nebyla schopna správně detekovat data, která ji ještě nikdy nebyly předloženy. Přeučení sítě by se mohlo projevit během trénovací fáze, kdy výsledky dosažené natrénovaným modelem by byly až příliš dobré, zatím co po předložení nezávislých dat by přesnost podstatně klesla.

Pokud disponujeme dostatečným množstvím dat, které lze následně rozdělit na tři dostatečně velké množiny, pak není problém použití jednoduchého přístupu a to vytvoření právě třetí validační množiny. Ve většině případech je však problematické vytvořit všechny tři množiny, právě kvůli nedostatku dat, a proto je zapotřebí přistoupit k využití některého ze sofistikovanějších přístupů validace. Jedním z nich je právě cross-validace, která je popsána v následující kapitole.

Cross-validace

Princip cross-validace spočívá rozdělení vstupních dat na sérii několika množin, kde se jedna podmnožina využije pro testování a všechny zbylé k trénování modelu. Tento proces se několikrát po sobě opakuje s různým rozdělením dat tak, aby pro testování byly předloženy vždy taková data, která nebyly použita pro trénování. Z každého opakování se počítá přesnost natrénovaného modelu, která je mnohdy reprezentována jako velikost odchylky. Následně se odchylky ze všech iterací zprůměrují a jejich výsledek reprezentuje celkovou cross-validační chybu.

Jedno z variant cross-validace je K-fold, neboli rozdělení na k množin, kde se pro trénování modelu využívají množiny *k-1*. Tento přístup je použit v této práci pro stanovení korektních výsledků při testování. Speciálním případem je rozdělení, kde k je rovno počtu záznamů v testovací sadě. Tento přístup je velice efektivní, ale při velkém množství dat se stává neúnosně časově náročný.

6.4 Výsledky

V této kapitole jsou prezentovány výsledky dosažené modelem neuronové sítě na datech, jenž byly uvedeny v kapitole 6.1. Podrobný rozpis dat v trénovací, testovací a validační množině je zobrazen v tabulce 6.6. Rozložení dat do jednotlivých množin zajišťovala funkce `train_test_split` z knihovny *Sklearn*, která v daném poměru (70:15:15) rozděluje všechny skupiny dat tak, aby se v každé množině nacházelo takové zastoupení skupin dat, které odpovídá poměru dělení tzn., že ve validační i testovací množině se nachází 15% od každého útoku i legitimního provozu.

Tab. 6.6: Zastoupení dat v jednotlivých množinách.

Druh provozu	Trénovací mn.	Testovací mn.	Validační mn.
Legitimní provoz	15307	3303	3274
SlowLoris	36	10	10
Rudy	83	18	14
SYN flood	320	69	79
UDP flood	321	64	83
NTP amplifikace	183	31	39
DNS amplifikace	192	36	25
Smurf	106	15	23

Počet neuronů ve výstupní vrstvě

Počet neuronů ve výstupní vrstvě jinými slovy udává, kolik výsledných tříd chceme rozeznávat. V tabulce 6.7 jsou reprezentovány výsledky dosažené neuronovou sítí, která měla ve výstupní vrstvě pouze jeden neuron, který rozhodoval, zda se jedná o útok nebo ne. Z výsledků je patrné, že bylo dosaženo téměř 100% úspěšnosti při identifikaci útoků i legitimního provozu. Z cross-validace (10-fold) vyplývá, že průměrná hodnota přesnosti určení útoku, je o 0.004 nižší než při použití testovací množiny pro získání výsledků.

Tab. 6.7: Úspěšnost modelu při detekci útoků a legitimního provozu.

Druh provozu	Přesnost [-]	Senzitivita [-]	F1_score[-]
Legitimní provoz	1,000	0,999	0,999
Útok	0,995	1,000	0,997
Cross-validace	0,991	0,998	0,998

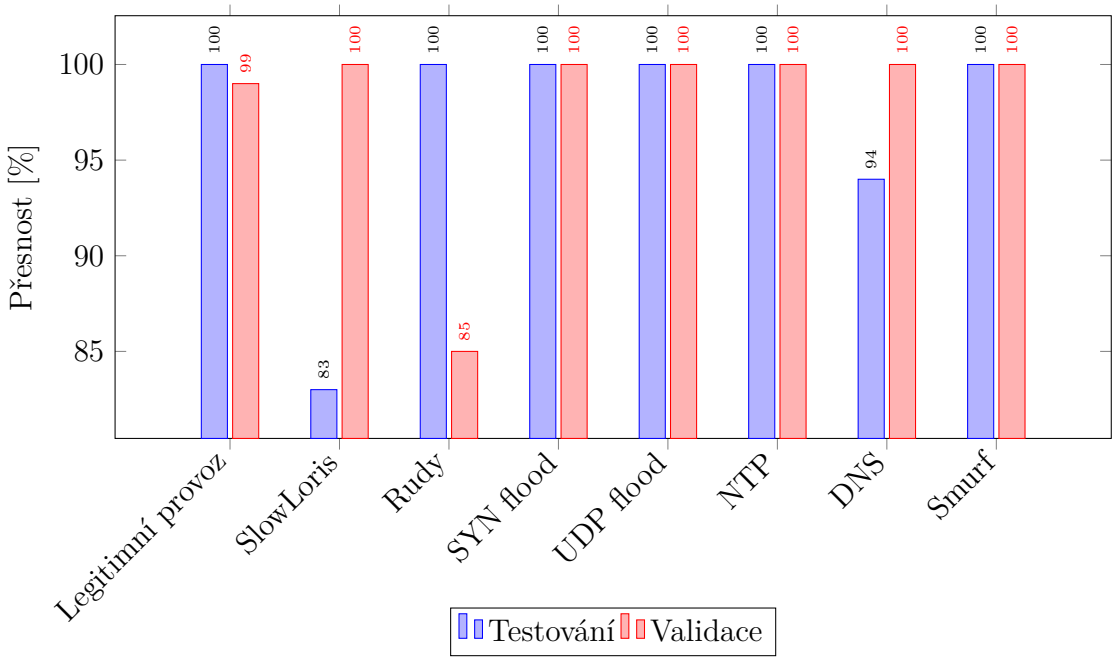
Pro testování neuronové sítě, která je schopna jmenovitě rozlišovat druhy provozu, je nutné, aby výstupní vrstva obsahovala více než jeden neuron. V kapitole 6.2.2 byla jako nejlepší aktivační funkce vybrána *ReLu*. Ta však nemůže být použita ve výstupní vrstvě neuronové sítě, proto musela být implementována do výstupní vrstvy funkce *Softmax*, která normalizuje všechny hodnoty, které ji přijdu na vstup a vytvoří z nich pravděpodobnostní rozložení pro `k`-tříd. V našem případě se `k` rovná počtu druhů detekovaných provozů. Výsledky dosažené touto neuronovou sítí jsou zobrazeny v tabulce 6.8. Tabulka ukazuje, že pouze pro útok SlowLoris a DNS amplifikaci nebyla přesnost označení 100%. Senzitivita legitimního provozu také nebyla 100%, avšak při velkém množství dat, které legitimní provoz běžně reprezentuje, se dá tenhle výsledek považovat za více než dostačující. Jedním z použitých ukazatelů bylo *F1_score*, které reprezentuje poměr mezi přesností a senzitivitou.

Od *F1_macro* je odlišuje pouze to, že v případě *F1_score* se jedná o hodnotu pro jednu třídu, ne pro souhrnný výsledek celé sítě.

Tab. 6.8: Výsledky pro jednotlivé testované skupiny.

Druh provozu	Přesnost [-]	Senzitivita [-]	F1_score [-]
Legitimní provoz	1,000	0,998	0,999
SlowLoris	0,833	1,000	0,909
Rudy	1,000	1,000	1,000
SYN flood	1,000	1,000	1,000
UDP flood	1,000	1,000	1,000
NTP amplifikace	1,000	1,000	1,000
DNS amplifikace	0,947	1,000	0,972
Smurf	1,000	1,000	1,000

Graficky jsou výsledky pro přesnost určení jednotlivých útoků v procentech zobrazeny na obr. 6.2, kde modré sloupce reprezentují výsledky dosažené na testovací množině a červené sloupce reprezentují výsledky dosažené na množině validační. Pro většinu provozů jsou výsledky stejné jak pro testování, tak i pro validování. Útoky SlowLoris a DNS amplifikace dosáhly horších výsledků při testování, zatím co Rudy při použití validační množiny. V obou případech mohlo dojít ke zhoršení výsledků z důvodu nedostatečné generalizace dosažené během trénování, kvůli tomu neuronová síť následně špatně určila typ provozu.



Obr. 6.2: Přesnost správného určení jednotlivých útoků.

Vliv jednotlivých parametrů na přesnost určení

Každý definovaný parametr určitou mírou přispíval k většímu množství kontrolovaných vlastností a napomáhal tak k určování provozu. Z tabulky 6.9 je patrné, že pokud dojde k odebrání jakéhokoliv parametru, tak ze sloupce zobrazujícího přesnost získanou cross-validací lze vidět, že všechny hodnoty jsou nižší než přesnost dosažena cross-validací, která je zobrazena v tabulce 6.7. Tudíž při odebrání jakéhokoliv parametru, dojde ke zhoršení detekčních schopností neuronové sítě.

Důvodem, proč právě při parametrech `Průměrná velikost payloadu` a `Průměrná hodnota TTL` byl pokles přesnosti nejvýznamnější může být více, ale hlavní důvod bude, že každý z nich dobře reprezentuje určitou skupinu útoků. Velikost payloadu ovlivní především útoky využívající TCP spojení, tudíž všechny použité *Low and slow* útoky. Na druhé straně `Průměrná hodnota TTL` bude významným markantem u takových záznamů, které nebudou poskytovat dostatek informací, využitelných při spočítání ostatních parametrů. Například útok Smurf, který je tímto parametrem významně ovlivněn.

Tab. 6.9: Úspěšnost bez jednoho parametru.

Vynechaný parametr	**Přesnost z CV[-]**
Celkový počet paketů	0,968
Počet otevřených TCP spojení	0,969
Průměrná velikost payloadu	0,837
Průměrná velikost paketu	0,977
Počet připojení hosta z různého portu	0,954
Nejvíce zastoupený protokol v %	0,950
Průměrná hodnota TTL	0,875

Souhrn výsledků

Výsledky, kterých bylo dosaženo, jsou ve většině případů převyšující hranici 90% přesnosti určení správného provozu, jenž dokazuje, že volené parametry dokázaly dostatečně dobře charakterizovat daný typ provozu v takové míře, aby to stačilo k natrénování neuronové sítě, která dokázala s vysokou pravděpodobností určit o jaký provoz se jedná. V tabulce 6.10 jsou výsledky získané pomocí cross-validací pro rozdílné množství neuronů ve výstupní vrstvě. Jak je vidět, tak pro detekci útoku a legitimního provozu neuronová síť dosahovala lepších výsledků, než pro neuronovou síť, která rozeznávala jednotlivé útoky. Na úkor většího množství možných výsledků, byla chybovost sítě s 8 neurony ve výstupní vrstvě větší, ale pouze v řádu setin.

Tab. 6.10: Porovnání výsledků výstupní vrstvy.

Počet neuronů	**1**	**8**
Přesnost [-]	0,995	0,984
Senzitivita [-]	0,998	0,998
F1_macro [-]	0,995	0,979

7 Závěr

Zadáním bakalářské práce bylo definování parametrů identifikujících síťové útoky a jejich implementace do navržené neuronové sítě, která by sloužila k detekci zvolených síťových útoků.

Teoretická část se zaměřuje na problematiku týkající se síťových anomálií a především metod jejich detekce. Další část je zaměřena především na strukturu a popis fungování umělé neuronové sítě. V neposlední řadě jsou v teoretické části rozebrány síťové útoky, které byly použity v rámci testování této bakalářské práce a krátce i popis jejich fungování. Poslední kapitola teoretické části se zabývá přípravou a zpracováním dat, před použitím v neuronové síti. Nejdůležitější částí teorie bylo vybrání vhodných parametrů, které by dokázaly charakterizovat zvolené síťové útoky a aby je na jejich základě bylo možné pomocí neuronové sítě detekovat.

Praktická část se zaměřuje především na sestavení efektivní neuronové sítě, která by sloužila k testování volených parametrů při detekci útoků. V první části jsou rozebrána data, která byla použita pro trénování, testování a následné validování neuronové sítě. Je popsána i metoda cross-validace, která sloužila k získání přesnějších výsledků, pomocí rozdělení trénovací množiny na několik menších trénovacích a testovacích podmnožin. Výsledkem cross-validace byla průměrná hodnota ze všech testovacích podmnožin. Tento výsledek poskytuje přesnější hodnotu, podle které můžeme usoudit, jak dobře by si natrénovaná neuronová síť vedla při označování dat, která ji nebyla nikdy předtím předložena. Důležitou podkapitolou praktické části je testování počtu neuronů ve skryté vrstvě a volba efektivní aktivační funkce. Tato část poskytla důležité informace pro nastavení neuronové sítě tak, aby při dalším testování bylo dosahováno nejlepších možných výsledků. Z testování vyšly nejlepší výsledky pro množství patnácti neuronů ve skryté vrstvě. Toto množství téměř odpovídá výsledku dosaženého vzorcem 6.2. Nejlepší aktivační funkcí se stala funkce *ReLu*, která během testování dosáhla nejlepších výsledků se všemi ukazateli úspěšnosti, avšak nejmenších časových nároků dosáhla funkce *Sigmoid*. Díky nízké časové náročnost by se dala uplatnit při zpracovávání velkého množství dat, kde by mohla dosahovat výsledků téměř o polovinu rychleji než ostatní aktivační funkce.

V poslední řadě se praktická část zabývá dosaženými výsledky. Testovány byly dvě varianty neuronové sítě. První varianta obsahovala pouze jeden neuron ve výstupní vrstvě, a proto rozpoznávala pouze legitimní provoz od útoku, ale útoky nedokázala dále identifikovat. Druhou variantou byla neuronová síť, která měla ve výstupní vrstvě 8 neuronů a dokázala tak nejen rozeznat legitimní provoz od útoku, ale dokázala i jednotlivé útoky identifikovat. Výsledky, kterých obě varianty dosahovaly, se pohybovaly nad hranicí 97% pro správné určení druhu provozu. Z toho lze vyvodit, že volené parametry poskytovaly dostatek informací neuronové síti, aby

dokázala různé provozy od sebe s vysokou pravděpodobností rozeznat.

Výstupem bakalářské práce je systém, který dokáže z předložené síťové komunikace detekovat útoky, které byly popsány v kapitole 4. Účinnost tohoto systému je dána především množinou trénovacích dat, které slouží jako základ pro detekci budoucích útoků. Data použita k trénování sítě byla laboratorně generována, a proto se nedají zcela srovnat s reálnou komunikací. Tento systém se dá použít jako jádro k vytvoření komplexnějšího systému, který by sloužil k detekci anomálií v síťovém provozu.

Literatura

[1] MELAMPY, Patrick. Networking anomalies. *NetworkWorld* [online]. 2018, 9.7.2018 [cit. 2018-11-28]. Dostupné z: https://www.networkworld.com/article/3284939/network-security/networking-anomalies.html

[2] BHUYAN, Monowar H., D. K. BHATTACHARYYA a J. K. KALITA. Network Anomaly Detection: Methods, Systems and Tools. IEEE Communications Surveys & Tutorials [online]. 2014, 16(1), 303-336 [cit. 2018-11-29]. DOI: 10.1109/SURV.2013.052213.00046.

[3] CHANDOLA, Varun, Arindam BANERJEE a Vipin KUMAR. Anomaly detection. *ACM Computing Surveys* [online]. 2009, 41(3), 1-58 [cit. 2019-04-04]. DOI: 10.1145/1541880.1541882.

[4] RIADI, Imam, Arif WIRAWAN a Sunardi -. Network Packet Classification using Neural Network based on Training Function and Hidden Layer Neuron Number Variation. *International Journal of Advanced Computer Science and Applications* [online]. 2017, 8(6) [cit. 2019-04-26]. DOI: 10.14569/IJACSA.2017.080631.

[5] KOC, Levent, Thomas A. MAZZUCHI a Shahram SARKANI. A network intrusion detection system based on a Hidden Naïve Bayes multiclass classifier. *Expert Systems with Applications* [online]. 2012, 39(18), 13492-13500 [cit. 2019-04-26]. DOI: 10.1016/j.eswa.2012.07.009.

[6] HEISELE, B., P. HO a T. POGGIO. Face recognition with support vector machines: global versus component-based approach. In: Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001 [online]. IEEE Comput. Soc, 2001, s. 688-694 [cit. 2019-04-26]. DOI: 10.1109/ICCV.2001.937693.

[7] KHAMPHAKDEE, Nattawat, Nunnapus BENJAMAS a Saiyan SAIYOD. Improving Intrusion Detection System based on Snort rules for network probe attack detection. In: *2014 2nd International Conference on Information and Communication Technology (ICoICT)* [online]. IEEE, 2014, 2014, s. 69-74 [cit. 2019-04-26]. DOI: 10.1109/ICoICT.2014.6914042.

[8] ZHANG, Ji a Hai WANG. Detecting outlying subspaces for high-dimensional data: the new task, algorithms, and performance. *Knowledge and Information Systems* [online]. 2006, 10(3), 333-355 [cit. 2019-04-05]. DOI: 10.1007/s10115-006-0020-z.

[9] WEI, Li, Weining QIAN, Aoying ZHOU, Wen JIN a Jeffrey X. YU. HOT: Hypergraph-Based Outlier Test for Categorical Data. WHANG, Kyu-Young, Jongwoo JEON, Kyuseok SHIM a Jaideep SRIVASTAVA, ed. *Advances in Knowledge Discovery and Data Mining* [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, 2003-4-30, s. 399-410 [cit. 2019-04-05]. Lecture Notes in Computer Science. DOI: 10.1007/3-540-36175-8 40.

[10] ŠORMOVÁ, Hana. *Detekce anomálií v datech o řešení problému.* Přírodovědecká fakulta, Masarykova univerzita, 2014. Diplomová práce. Masarykova univerzita. Vedoucí práce Doc. Mgr. Radek Pelánek, Ph.D.

[11] BASU, Sugato, Mikhail BILENKO a Raymond J. MOONEY. A probabilistic framework for semi-supervised clustering. In: *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04* [online]. New York, New York, USA: ACM Press, 2004, 2004, s. 59- [cit. 2019-04-05]. DOI: 10.1145/1014052.1014062.

[12] BIVENS, Alan, Chandrika PALAGIRI, Rasheda SMITH a Boleslaw SZYMANSKI. Clustering approaches for anomaly based intrusion detection. *Proceedings of the Intelligent Engineering Systems through Artificial Neural Networks.* ASME Press, 2002(vol. 12), 579-584.

[13] OTEY, M., S. PARTHASARATHY, A. GHOTING, G. LI, S. NARRAVULA a D. PANDA. Towards NIC-based intrusion detection. In:*Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '03* [online]. New York, New York, USA: ACM Press, 2003, 2003, s. 723- [cit. 2019-04-05]. DOI: 10.1145/956750.956847.

[14] ZHANG, Weiyu, Qingbo YANG a Yushui GENG. A Survey of Anomaly Detection Methods in Networks. In: *2009 International Symposium on Computer Network and Multimedia Technology* [online]. IEEE, 2009, 2009, s. 1-3 [cit. 2018-11-30]. DOI: 10.1109/CNMT.2009.5374676.

[15] VONDRÁK, Ivo. *Umělá inteligence a neuronové sítě.* Dot. Ostrava: Vysoká škola báňská - Technická univerzita Ostrava, 1998. ISBN isbn80-707-8259-5.

[16] VOLNÁ, PHD, RNDr. PaedDr. Eva. *Neuronové sítě 1.* Druhé. Ostrava: Ostravská univerzita v Ostravě, 2008. Studijní materiál. Ostravská univerzita v Ostravě.

[17] ULDRICH, Miloš a Tomáš JURCZYJ. *Neuronové sítě a jejich využití.* IT Systems. 2014, 2014(3), 2-3.

[18] KUPREEV, Oleg, Ekaterina BADOVSKAYA a Alexander GUTNIKOV. DDoS Attacks in Q4 2018. *AO Kaspersky Lab* [online]. Kaspersky Lab, 2019 [cit. 2019-03-11]. Dostupné z: https://securelist.com/ddos-attacks-in-q4-2018/89565/

[19] ALI, Siti Hajar Aminah, Seiichi OZAWA, Tao BAN, Junji NAKAZATO a Jumpei SHIMAMURA. A neural network model for detecting DDoS attacks using darknet traffic features. *2016 International Joint Conference on Neural Networks (IJCNN)* [online]. IEEE, 2016, 2979-2985 [cit. 2018-11-12]. DOI: 10.1109/IJCNN.2016.7727577.

[20] HSIEH, Chang-Jung a Ting-Yuan CHAN. Detection DDoS attacks based on neural-network using Apache Spark. *2016 International Conference on Applied System Innovation (ICASI)* [online]. IEEE, 2016, 1-4 [cit. 2018-11-12]. DOI: 10.1109/ICASI.2016.7539833.

[21] SHAH, Bhavin a Bhushan H. TRIVEDI. *Reducing Features of KDD CUP 1999 Dataset for Anomaly Detection Using Back Propagation Neural Network* [online]. IEEE, 2015, 2015, , 247-251 [cit. 2018-11-12]. DOI: 10.1109/ACCT.2015.131.

[22] The CTU-13 Dataset. A Labeled Dataset with Botnet, Normal and Background traffic. *Stratosphere IPS* [online]. Praha: CTU University, 2011 [cit. 2018-12-02]. Dostupné z: https://www.stratosphereips.org/datasets-normal/

[23] Slowloris HTTP DoS attack in Python. *GitHub* [online]. 2015 [cit. 2018-12-02]. Dostupné z: https://github.com/adrianchifor/pyslowloris

[24] RUDY. *GitHub* [online]. 2015 [cit. 2018-12-02]. Dostupné z: https://github.com/nosperantos/RUDY

[25] HUNTER, David, Hao YU, Michael S. PUKISH, III, Janusz KOLBUSZ a Bogdan M. WILAMOWSKI. Selection of Proper Neural Network Sizes and Architectures—A Comparative Study. IEEE Transactions on Industrial Informatics [online]. 2012, 8(2), 228-240 [cit. 2018-12-07]. DOI: 10.1109/TII.2012.2187914.

[26] StackExchange. *How to choose the number of hidden layers and nodes in a feedforward neural network?* [online]. [cit. 4. 12. 2017]. Dostupné z URL: <https://stats.stackexchange.com/questions/181/how-to-choose-the-number-ofhidden-layers-and-nodes-in-a-feedforward-neural-netw/1097#1097>.

[27] DAUBNER, Lukáš. *Deep learning.* Brno, 2015. Bakalářská práce. Masarykova univerzita. Vedoucí práce Doc. RNDr. Lubomír Popelínský, Ph.D.

Seznam symbolů, veličin a zkratek

DDoS	Distributed Denial of Service – distribuované odepření služby
IoT	Internet of Things – internet věcí
IDS	Intrusion Detection System – systém pro odhalování průniků
SVM	Support Vector Machine – metoda strojového učení s učitelem
IP	Internet Protocol – protokol sloužící pro směrování v počítačových sítích
ICMP	Internet Control Message Protocol – protokol sloužící k ověřování dostupnosti zařízení
HTTP	Hypertext Transfer Protocol – internetový protokol sloužící k přenosu hypertextových dokumentů
TCP	Transmission Control Protocol – spojově orientovaný protokol transportní vrstvy modelu ISO/OSI
UDP	User Datagram Protocol – nespolehlivý protokol transportní vrstvy
TP	True positive – správné označení positivní informace
TN	True negative – správné označení negativní informace
FP	False positive – špatné označení positivní informace
FN	False negative – špatné označení negativní informace
VM	Virtual Machine – virtuální stroj
VoIP	Voice over Internet Protocol – technologie, která umožňuje přenos digitalizovaného hlasu
NTP	Network Time Protocol – protokol zajišťující synchronizaci hodin počítačů
DNS	Domain Name System – hierarchický systém doménových jmen
BFGS	Iterativní metoda řešení nelineárních optimalizačních problémů
TTL	Time To Live – hodnota určující maximální dobu existence IP datagramu